Aalto University
School of Science
Degree Programme in Computer Science and Engineering

Juhana Kommeri

Computer-aided design software for custom nucleic acid nanostructures

Master's Thesis
Espoo, October 2, 2016

Supervisors: Professor Mauri Kostiainen
Advisor: Ph.D., Docent Veikko Linko

Aalto University
School of Science
Degree Programme in Computer Science and Engineering

ABSTRACT OF
MASTER'S THESIS

Author: Juhana Kommeri

Title:
Computer-aided design software for custom nucleic acid nanostructures

Date: October 2, 2016 **Pages:** 72

Major: Media technology **Code:** T-111

Supervisors: Professor Mauri Kostiainen

Advisor: Ph.D., Docent Veikko Linko

The past decade has witnessed significant improvement in the techniques for assembling nucleic acid molecules into desired nanostructures. Today, structural DNA nanotechnology is seen as an attractive field by many multidisciplinary researchers who are impressed on the ability to position and control molecules with nanometer-level accuracy. The rationally designed DNA structures possess a tremendous potential in many bionanotechnological applications, and therefore the complexity of the designed structures is constantly increasing. To follow this trend and to enable straightforward design of such structures, powerful and versatile computer-aided design software and simulation tools are urgently needed.

In this thesis various design software for creating nucleic acid nanostructures are assessed against predefined criteria. Using these software, exemplary DNA designs are created, and the features of the software are described. The aim of this comparison is to give a comprehensive overview of the currently available software and help a reader to decide which software to use for each particular purpose.

In addition, a specific plugin for the most commonly used software caDNAno is created. By using this plugin, a hybrid RNA-DNA nanostructure is designed. Along with this and the simulation of the structure, the functionality of the plugin is evaluated. This simple plugin is created in order to speed up the design process of such structures, and here, it is used to produce sequences of the DNA strands that are needed for the abovementioned structure. Moreover, the plugin demonstrates the possibility to extend the available software and their features, and it serves as an example for any user to create their own custom plugins and to integrate them to the software.

Keywords: nucleic acids, DNA nanotechnology, DNA origami, CAD

Language: English

Aalto-yliopisto
Perustieteiden korkeakoulu
Tietotekniikan koulutusohjelma

DIPLOMITYÖN TIIVISTELMÄ

Tekijä:	Juhana Kommeri		
Työn nimi:	Tietokoneavusteiset suunnitteluohjelmat nukleiinihapponanorakenteille		
Päiväys:	2. lokakuuta 2016	**Sivumäärä:**	72
Pääaine:	Mediatekniikka	**Koodi:**	T-111
Valvojat:	Professori Mauri Kostiainen		
Ohjaaja:	FT, dosentti Veikko Linko		

Tekniikat nukleiinihappoihin perustuvien nanorakenteiden valmistuksessa ovat kehittyneet merkittävästi viimeisen vuosikymmenen aikana. Tällä hetkellä rakenteellinen DNA-nanoteknologia nähdään yleisesti kiinnostavana tieteenalana, sillä se mahdollistaa molekyylien järjestämisen ja asemoinnin nanometrin mittakaavassa. Ohjelmoiduilla DNA-rakenteilla on mahdollista toteuttaa monia potentiaalisia sovelluksia lähitulevaisuudessa, ja niinpä suunnitellut nanorakenteet ovat muodostuneet koko ajan monimutkaisemmiksi. Jotta tutkijat voisivat jatkossakin tehdä kompleksisia rakenteita sovelluksia varten, tarvitsevat he yhä tehokkaampia ja monipuolisempia tietokoneavusteisia ohjelmia rakenteiden suunnitteluun ja simulointiin.

Tässä diplomityössä erilaisia käytössä olevia suunnitteluohjelmia verrataan keskenään tiettyjen arviointikriteerien perusteella. Kyseisiä ohjelmia käytetään myös DNA-pohjaisten esimerkkirakenteiden suunnitteluun, ja prosessin perusteella listataan ohjelmien ominaisuuksia. Työn tavoitteena on antaa saatavilla olevista ohjelmista ja niiden ominaisuuksista kattava yleiskuva, jotta lukija voisi valita itselleen sopivan ohjelman halutun nanorakenteen suunnitteluun.

Lisäksi tässä työssä kehitetään lisäosa caDNAno-sovellukseen, joka on yksi käytetyimmistä ohjelmista nukeliinihapporakenteiden suunnittelussa. Tätä lisäosaa käyttäen työssä suunnitellaan RNA-DNA–hybridinanorakenne. Suunnitteluprosessin ja simulaation avulla varmistetaan lisäosan toimivuus. Yksinkertainen lisäosa nopeuttaa kyseisten rakenteiden suunnittelua ja tässä työssä lisäosan avulla määritetään tarvittavien DNA-juosteiden sekvenssit. Lisäosan suunnittelu toimii myös esimerkkinä sovellusten laajennettavuusmahdollisuuksista, ja kyseinen esimerkki voi auttaa käyttäjiä omien kustomoitujen lisäosien suunnittelussa ja integroimisessa sovellukseen.

Asiasanat:	nukleiniihapot, DNA-nanoteknologia, DNA-origami, CAD
Kieli:	Englanti

Acknowledgements

I would like to thank my supervisor Prof. Mauri Kostiainen for giving me an opportunity to carry out this Master's Thesis project in his research group. Special thanks are devoted to my advisor Docent Veikko Linko for his continuous support and guidance throughout the project. Without him this thesis would not have been possible to accomplish.

Espoo, October 2, 2016

Juhana Kommeri

Abbreviations and acronyms

A-DNA	one of the major forms of dsDNA
AFM	atomic force microscopy
B-DNA	one of the major forms of dsDNA, the 'standard' form
bp	base pair
CAD	computer-aided design
cryo-EM	cryo-electron microscopy
CSV	comma-separated values
DX	double-crossover
DNA	deoxyribonucleic acid
dsDNA	double-stranded DNA
dsRNA	double-stranded RNA
HB	helix bundle
IC	integrated circuit
Incl	inclination angle
NMR	nuclear magnetic resonance
nt	nucleotide
P-Tw	propeller twist
PEG	polyethylene glycol
PLY	polygon file format
RNA	ribonucleic acid
siRNA	small interfering RNA or silencing RNA
ssDNA	single-stranded DNA
TEM	transmission electron micropscopy
x-Dis	x-displacement
Z-DNA	one of the major forms of dsDNA
60HB	60-helix bundle
64HB	64-helix bundle

Contents

Chapter 1

Introduction

The main purpose of DNA and RNA nanotechnology is to create artificial rationally designed nucleic acid nanostructures for diverse applications in biology, chemistry and physics. DNA and RNA are well-known for their essential roles in all living systems: DNA serves as the carrier of genetic information, whereas RNA molecules can have several tasks including decoding the genetic information and catalyzing reactions. However, in the field of nanotechnology, DNA molecules can be utilized as a high-capacity information storage [14, 32, 111], and interestingly, both DNA and RNA molecules can be used as construction material to assemble accurate, biocompatible and functional nanostructures [36, 41, 62, 84]. At present, an ever-increasing number of research groups are exploiting programmable self-assembly properties of nucleic acids in creating rationally designed nanoshapes and nanomachines for many different uses. [1, 41] In this thesis the main focus is on DNA nanostructures, since DNA is more rigid, stable and easily controllable molecule than RNA, and therefore it is more common choice when custom nucleic acid nanostructures are fabricated. However, most of the design principles presented here for DNA can be equally used for RNA-based nanostructures.

DNA nanotechnology, or more specifically 'structural DNA nanotechnology', has grown rapidly over the past 30 years. Along with the growth and increasing complexity of the nanostructures the need for versatile design software has arisen. Nowadays there exist several user-friendly solutions to design and predict different conformations of nucleic acid nanostructures. The purpose of this thesis is to give a comprehensive overview of the available software for designing such structures. In this thesis, pros and cons of each program are listed, and the software are compared against each other. The specific features of each program are introduced, and in the end, the aim is that a reader have gained a good understanding on which software to choose for designing a desired nucleic acid nanoshape. In addition, a simple

plugin for the most used DNA software called caDNAno [21] is created. The purpose of the plugin is to show how the program can be extended, and on the other hand, the plugin can be used to simplify the design process for example hybrid RNA-DNA nanostructures that require different rules than simple DNA-DNA structures.

In Chapter 2 some essential background is given in order to understand the nature of nucleic acids and their properties that are exploited in assembling nanostructures. Moreover, the brief history of DNA nanotechnology is covered and the already existing and imminent applications are shortly discussed. In Chapter 3 the overall design process and the design principles of DNA nanostructures are explained in detail, and the rationale to design software is given. Chapter 4 is the most important one including short introductions to the currently available programs, designs of exemplary structures using each program, and finally, a result section with summary and comparison of the programs. In Chapter 5 a specific plugin for caDNAno is implemented, and in Chapter 6 the created plugin is evaluated by designing a 2D and 3D RNA-DNA nanostructures with it. Finally, in Chapter 7 the main results are analyzed and discussed.

This thesis project is carried out in Mauri Kostiainen's research group, which has a strong background in the field of experimental DNA nanotechnology. Their very recent work includes for example the technique to controllably form DNA nanostructures using external signals [26], implementations for positioning DNA origamis on different substrates [48, 67, 92], and metallization of various DNA origami shapes [25, 93]. This part of the research deals with materials science, and the results could be used in developing DNA-based nanoelectronics [68] and DNA-templated nanoplasmonics [94]. Currently, the group is using DNA origamis as cell-delivery vehicles [47, 71, 73], and along these lines, they are aiming toward DNA-based medical devices [66] and fully biocompatible nanoreactors/sensors [63, 65]. Therefore, this thesis could serve as a guide for the group's new students who would need to select the suitable design software when creating novel DNA/RNA nanostructures for the above-mentioned applications. Moreover, the extension plugin designed in this work (and other similar plugins) could be readily included into the caDNAno design software if needed.

Chapter 2

Background

There are four characteristic building blocks (bases) in deoxyribonucleic acid (DNA) molecules: adenine, guanine, cytosine and thymine. In DNA nanotechnology, DNA molecules are assembled *via* specific Watson-Crick base pairing [100] into pre-defined and programmed structures. In general, the dimensions of the structures and devices used in nanotechnology are usually below 100 nanometers, and most of the DNA structures fit easily into that volume. Importantly, assembling structures using DNA molecules may enable even sub-nanometer patterning resolution, thus making DNA nanostructures ideal candidates for many practical uses in the 'nanorealm' [28]. Unfortunately, many implementations of DNA nanotechnology are still quite impractical, but as the field is growing fast, the diverse applications may arise rapidly. For example, there are already applications to help researchers map the atomic structure of proteins, computing in living cells, and soon, tracking and curing diseases with DNA structures might become possible. [66, 75, 91] In this chapter the properties of nucleic acids are presented. In addition, it is described how the self-assembly process works in practice. Furthermore, the evolution of the field and some key (current and future) applications are briefly discussed.

2.1 Properties of nucleic acids

The foundation of the nucleic acid nanotechnology is molecular self-assembly: the process in which the molecules arrange themselves into stable and predefined structures. Nucleic acid is a polymer, and it is comprised of nucleotides, which contain a phosphate group, a sugar group and a nitrogen base as shown in Figure 2.1. The nitrogen bases in deoxyribonucleic acid (DNA) are called adenine (A), thymine (T), guanine (G) and cytosine (C). In the

case of ribonucleic acid (RNA), the thymine is replaced with uracil (U). [79]

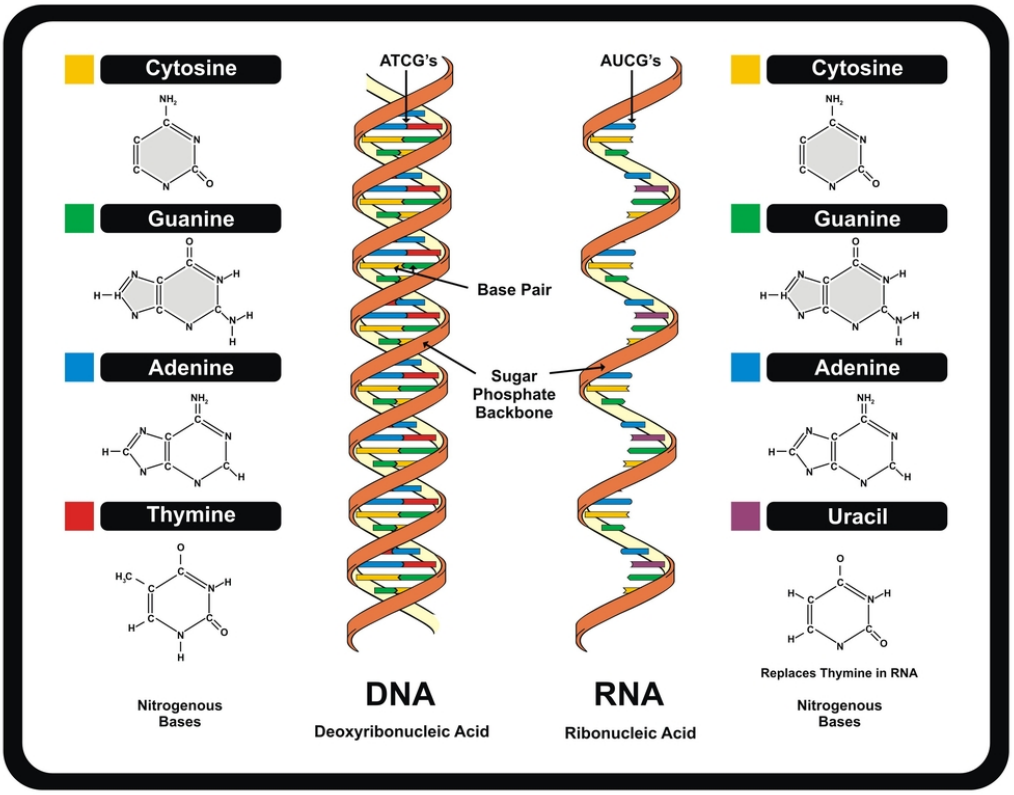

Figure 2.1: The structure of DNA and RNA [79].

DNA strands are formed by the backbone of repeating sugar-phosphate units (note that individual sugars and phosphates are not depicted in Figure 2.1) that contain aforementioned bases. DNA strands can be attached to each other by hydrogen bonds, thus forming a helical structure called double helix (the diameter of the double helix is about 2 nm [87]). In detail, the double helix is formed when two anti-parallel nucleic acid strands are attached to each other by following strict base pairing (bp) rules. In the case of DNA, adenine (A) is attached to thymine (T) and cytosine (C) is attached to guanine (G). RNA is similar, but adenine (A) is attached to uracil (U). These pairing rules are usually referred as Watson-Crick base pairing rules [100]. By exploiting Watson-Crick base pairing rules, it is possible to fabricate self-assembling structures and machines with nucleic acid geometries not found from the Nature. [1, 5]

Modern biotechnology methods are able to produce long synthetic DNA

molecules, where a sequence of bases can be arbitrarily chosen. This allows us to design strands of DNA, which will self-assemble into complex forms. A chemical system containing the designed strands tends to change to a state that has the lowest free energy. The free energy is minimized when two complementary (Watson-Crick base pairing rules) strands of nucleotides form a double helix, and therefore nucleic acid strands will form a conformation where the number of correctly paired bases is maximized. [85] An example of a self-assembly process is illustrated in Figure 2.2, where a long DNA strand is 'folded' by multiple short strands. This method is called 'DNA origami', one of the most important techniques used in DNA nanotechnology. The different design methods are discussed in more detail in Chapter 3.

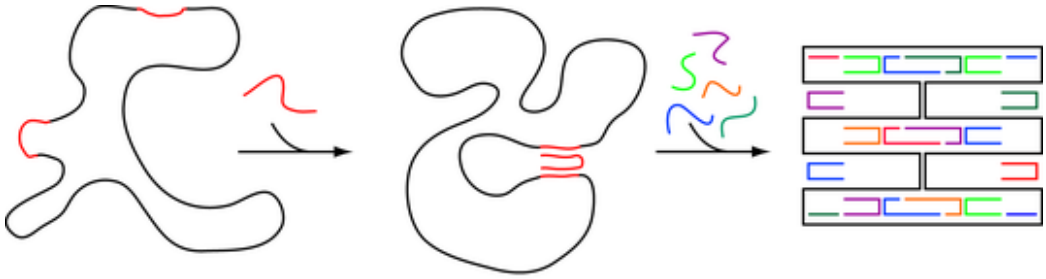

Figure 2.2: An example of a self-assembly process, dubbed 'DNA origami'. All colored staple strands find the binding locations (red strand binds to red segments) at the long scaffold strand (black) and thus the scaffold strand is folded into a pre-defined shape [57].

Depending on the environmental conditions, the double-helical DNA can have different forms. Three of the most common forms are A-, B- and Z-DNA, where B-DNA is the standard form in the cell and it was for a long time considered as the only biologically relevant structure. When X-ray diffraction photographs of DNA fibers became available, it was clear that there exists at least two different forms of double helices: A-form and B-form. In low humidity environment DNA adopts A-form and respectively in higher humidity it becomes B-form. Currently, there are different DNA structures associated for almost every English alphabet excluding only F, Q, U, V and Y letters. Some of those conformations differ only slightly from another, whereas some are completely different. For example, the handedness, the number of strands or the base-pairing scheme may differ. Figure 2.3 describes the differences between A-, B- and Z-DNA models. A- and B-DNA have right-handed helices and the helix diameter of A-DNA is about 2.6 nm. A-DNA has about 11 bp per helical turn and it rises approximately 0.23 nm per

bp along axis. The diameter of B-DNA is about 2 nm, it has 10.5 bp per turn and it rises about 0.34 nm per bp. Z-DNA, for one, is a left-handed helix and the corresponding parameters are 12 bp per turn and helix rise 0.38 nm. The inclination or tilt of bp to axis is approximately +21° in A-DNA, -6° in B-DNA and in Z-DNA the angle is -6°. B-DNA is the most stable form of a DNA molecule and its conformation is used as the basis for designing different nanostructures. [6, 39, 61]

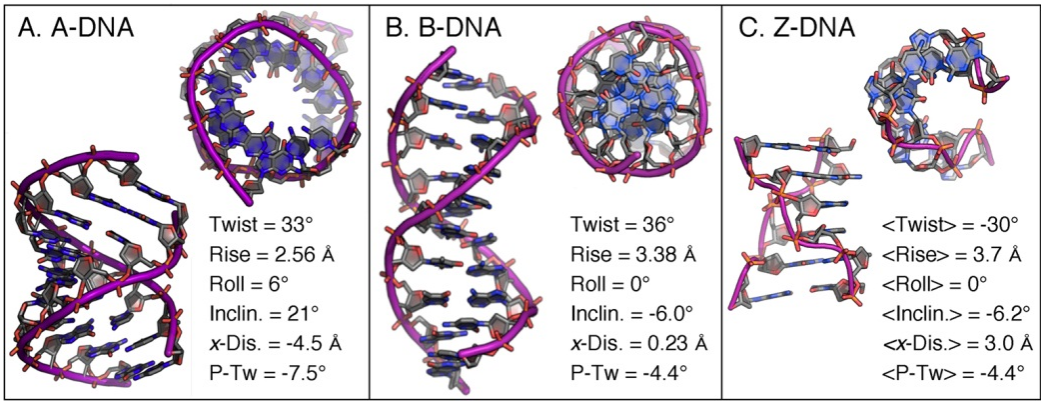

Figure 2.3: Structures of the major forms of double-stranded DNA: A-DNA, B-DNA and Z-DNA [39]. Incl. = inclination angle, x-Dis. = x-displacement, P-Tw = propeller twist.

The design of DNA nanostructures is usually based on the assumption that DNA adopts a natural B-form in the self-assembly process. However, the end result may be different, as stated in the study by Bai *et al.* [5]. This was the first study to validate the assumption that the positioning of structural elements in DNA nanostructure could be controlled with subnanometer precision. They designed a test object suitable for low-temperature electron microscopy (cryo-EM) and found out that the helices follow three-dimensional chickenwire-like pattern, which bends away from connecting crossovers both in horizontal and vertical directions. Crossovers (Holliday junctions) connecting the adjacent helices have unusual geometry and their positions are not precisely at 270° angles with respect to helical axis, causing the structure to twist (see Chapter 4 for the explanation). There exist unusual motifs, so called left-handed pseudohelices, which have no natural equivalent and where the base pairs are oriented along the helical axes rather than perpendicularly as in the standard B-form DNA. The complex adapts a local minimum energy state, and the final conformation of the structure is a sum of several aspects. Overall, it is a technological challenge to position atoms as precisely

as enzymes do, but interestingly, this study showed that such precision is achievable using synthetic rationally designed object made of DNA.

2.2 A brief history of DNA nanotechnology

Nadrian 'Ned' Seeman is regarded as the founder of the DNA nanotechnology. In 1980s he was trying to figure out how to use DNA as construction material instead of as a genetic control. The original idea of structural DNA nanotechnology was to utilize three-dimensional DNA lattices in organizing molecules for crystallography (see Fig. 2.4(b)).

In the 1990s the historically important double crossover (DX) molecule was first reported, and in such a molecule / molecular bundle two DNA double helices are fused together via crossovers [29]. The DX motif was used to create robust periodic lattices *via* sticky-ends (short single-stranded DNA overhangs protruding from the structure), and this approach serves as the basis for the first 2D arrays. [88, 104] Figure 2.4(a-c) illustrates these kind of self-assembled arrays: (a) shows a branched DNA building block with four distinct arms, (b) is a schematic view of the 3D crystalline DNA lattice with proteins organized into it and in (c) there are 2D and 3D lattices constructed from small repeating DNA nanostructure motifs.

One of the breakthroughs of this kind of assembly was the 'DNA cube' where the positions of each atom were programmed and known. The cube was actually the first rationally designed three-dimensional nanoscale DNA object and it brought Seeman the Feynman Prize in Nanotechnology in 1995. [91] Such polyhedral DNA structures are presented in Figure 2.4(d). In addition, two-dimensional tile -based structures enable algorithmic assembly such as Sierpinski triangles [81] (see Fig. 2.4(e)).

The abovementioned 'building block' approaches are quite ineffective, because the number of successfully assembled structures might be low and it is also hard to limit the overall size of those 2D lattices [61]. In 2006 the DNA origami technique [80] was first demonstrated by Paul Rothemund, and it was a revolutionary technique for building more complex and better controlled structures. In this technique a long scaffold strand is folded by shorter 'staple' or 'helper' strands into a well-defined shape [80, 88]. Figure 2.4(f) demonstrates some of the nanoshapes that were created using this method. Figure 2.2 illustrates the self-assembly process of the DNA origami. Usually the long strand used is a genome of a M13 virus, because it is commercially available and it has a proven track record as a suitable scaffold strand [62].

In general, DNA origami is a straightforward method to build custom, complex shapes with dimensions below 100 nm. Spatial accuracy of such

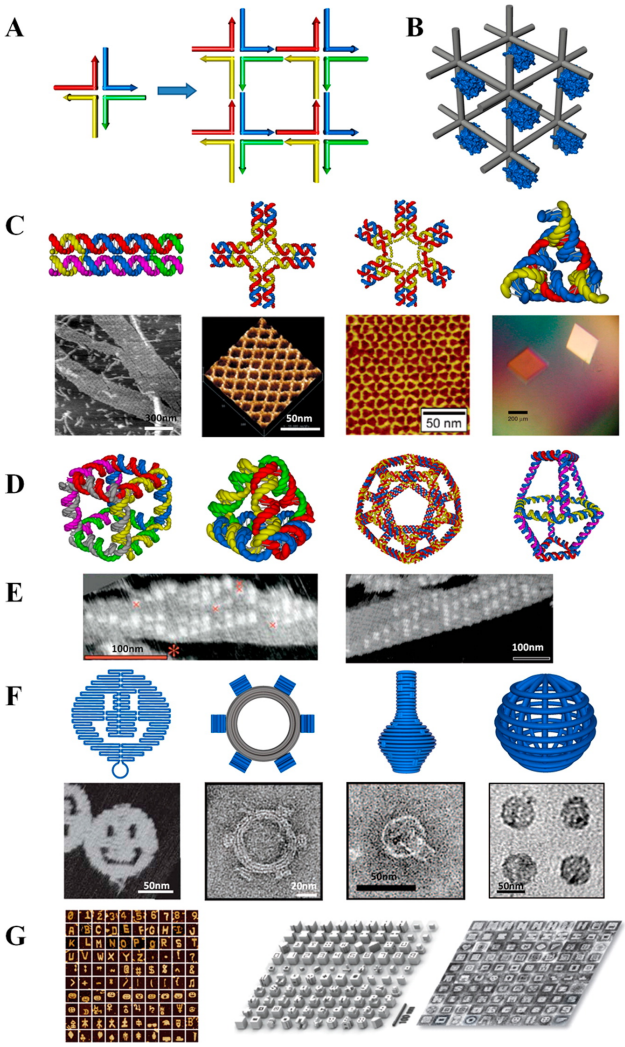

Figure 2.4: Foundations of structural DNA nanotechnology [109]. (a) Original proposal by Seeman to use immobile junctions to create self-assembling arrays. (b) The idea to use DNA crystalline lattices to organize macromolecules. (c) Periodic 2D and 3D crystals created by DNA nanostructure motifs. (d) Polyhedral DNA nanostructures. (e) Sierpinski triangles and a binary counter. (f) 2D and 3D DNA origami nanostructures. (g) Complex nanostructures based on single-stranded DNA tiles.

objects can reach subnanometer scale [5]. The key feature of DNA origami technique is the ability to assemble materials at the nanoscale with enormously high precision. There are numerous examples of positioning DNA

origamis on the various substrates and assembling *e.g.* proteins and metal nanoparticles accurately using the DNA origami templates. These structures could find uses in many applications (see the next section). [16, 42, 45, 50, 58, 65, 68, 70, 92]

Original two-dimensional DNA origami technique has been later generalized to 3D fabrication based on either 2D origami sheets [4] or multilayer structures [21]. In addition, it is possible to create 3D structures with custom curvatures, twists and bends [17, 37] (Fig. 2.4(f)) as well as prestressed tensegrity structures [56]. Importantly, DNA origami has served as an excellent starting point for the more recent design techniques such as 'scaffold-free origami' or 'single-stranded tile/brick' assembly [44, 101] (Fig. 2.4(g)) and the various wireframe-based meshing techniques [9, 10, 38, 40, 98] (see more details in Chapter 4). The approach that is based on single-stranded assembly (instead of scaffolding) has recently enabled genetic encoding -based production of simple DNA shapes in living bacteria cells [23].

Figure 2.5 shows how the interest toward structural DNA nanotechnology has grown over the past decades along with some of the key structures and the year they were published. The number of cumulative citations in the low panel shows that the field is enjoying rapid growth and that interest towards the field is constantly increasing.

In addition, it should be mentioned here that instead of DNA, RNA could be equally used in fabrication of nanostructures [30, 35, 36]. Moreover, there exist several examples of RNA-DNA hybrid structures [24, 99] (a plugin for designing RNA-DNA hybrid using caDNAno is presented in Chapters 5 and 6). However, so far RNA has not been used as widely as DNA in nanofabrication due to its weaker mechanical properties. Moreover, the currently available RNA structures are limited in size. However, RNA has some interesting features, for example that it can be folded (with itself) during transcription and thus the target shapes could be genetically encoded and expressed in living cells [30]. RNA-based or RNA-DNA hybrid structures could find uses in delivery applications owing to their important functions. For example, double-stranded RNA (dsRNA) or small interfering / silencing RNA (siRNA) can induce RNA interference in cells. It has been showed that folate- or peptide-modified tetrahedral DNA nanoparticles loaded with siRNAs can silence target genes in tumors [55]. This is a neat example of the possibilities provided by the rational design of DNA-RNA nanostructures: the delivery of siRNAs strongly depends on the density and the spatial orientation of the cancer-targeting ligands, and they can be precisely tuned using DNA-based nanoparticles.

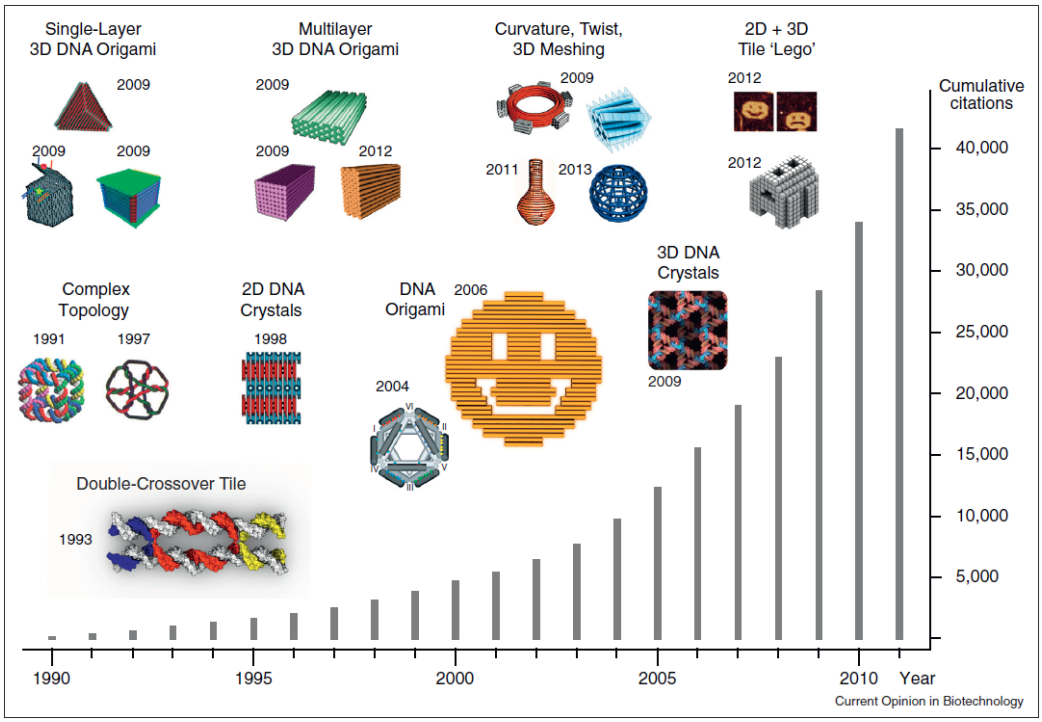

Figure 2.5: Some key structures and techniques in structural DNA nanotechnology. The lower panel depicts the growth of interest toward the field: the histogram shows cumulative citation count to a selected set of articles. [62].

2.3 Applications

DNA nanotechnology can be divided into two subfields: structural and dynamic DNA nanotechnology. In structural DNA nanotechnology the objects are static, meaning that the end state of an assembly will be at equilibrium. Examples of assemblies in structural DNA nanotechnology are DNA nanowires, nanorods and DNA structures decorated with quantum dots and metal nanoparticles [109]. Dynamic DNA nanotechnology, for one, aims to create structures that are non-static, and will transform based on chemical or physical conditions. Examples of dynamic complexes are circuits, catalytic amplifiers, autonomous molecular motors and re-configurable nanostructures. [86, 107]

In addition, DNA nanostructures can find interesting applications in characterization and imaging of other biomolecules; for example nucleic magnetic resonance (NMR)-based characterization (Fig. 2.6(a) [20], AFM-frames (Fig. 2.6(b) [82]) and templates for cryo-electron microscopy (Fig. 2.6(c),

[89]) have been already demonstrated. Moreover, DNA origami designs have been utilized as smart lids on the solid-state nanopores, which could be used in state-of-the-art sensing and sequencing applications (Fig. 2.6(d) [8, 76, 102]). On top of that, DNA origamis can be exploited in creating plasmonic structures (Fig. 2.6(f) [50, 51, 93]), and rulers for optical high-resolution nanoimaging (Fig. 2.6(g) [42, 58]). The structural addressability, molecular scale resolution and the high parallelity that can be achieved using DNA nanostructures could be utilized in novel nanofabrication techniques (combined top-down and bottom-up approaches), and this may lead to implementations in molecular electronics and plasmonics and in creating integrated circuits (ICs) and metamaterials [33, 34, 45, 68, 70, 92, 93].

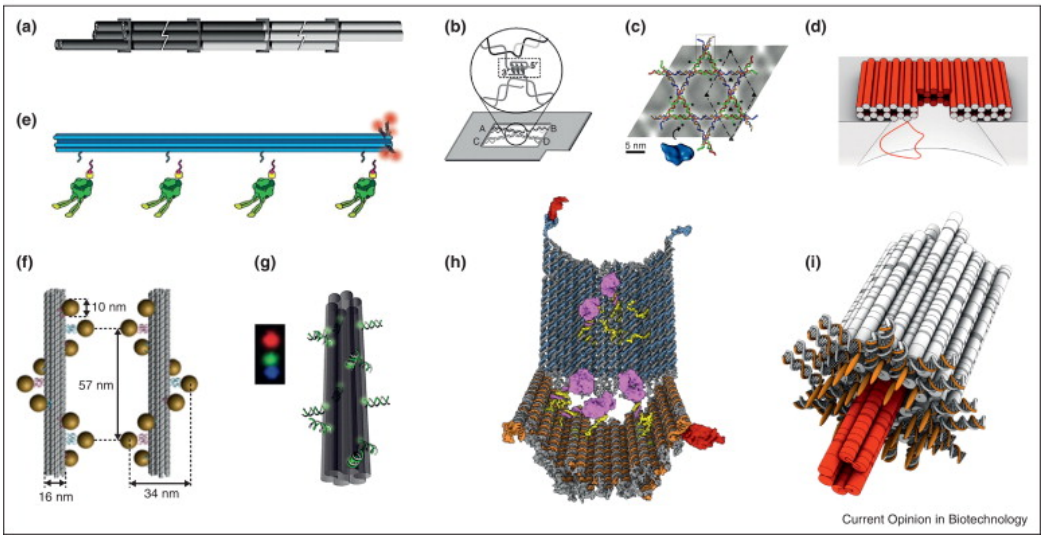

Figure 2.6: Applications provided by DNA nanotechnology [62]. (a) DNA nanotubes for nuclear magnetic resonance (NMR) -based characterization of proteins [20]. (b) An atomic force microscopy (AFM) frame for studying molecular switching of G-quadruplex [82]. (c) DNA crystals for organizing and imaging proteins using (cryo-)electron microscopy [89]. (d) DNA origami gatekeepers/lids for solid-state nanopores [102]. (e) DNA origami acts as a cargo for motor proteins [16]. (f) Plasmonic nanonostructures: DNA origami decorated with nanoparticles in a chiral fashion [50]. (g) DNA origami-based fluorescent barcodes for optical microscopy [58]. (h) Smart DNA origami nanorobot for drug-delivery [19]. (i) A DNA origami-based artificial ion channel for lipid membranes [54].

Some of the useful and innovative applications are about to emerge in

the medicinal field [66], such as DNA nanostructures that can be used as vehicles and devices for targeted drug delivery (Fig. 2.6(h) [19]) and molecular transport (Fig. 2.6(e) [16] and Fig. 2.6(i) [49, 54]), or as nanorobots that are able to interact with each other and perform logic computations in living organisms [2]. In addition, due to its superior self-assembly and programming properties, DNA is a truly interesting molecule in molecular computation. DNA molecules and higher-order nanostructures can be used in algorithmic self-assembly and digital computing [7] and for example, artificial neural networks can be created at molecular level using DNA strands [77, 78].

Chapter 3

Design principles of DNA nanostructures

Currently there exist several user-friendly methods and software to design and predict the 3D conformation of DNA nanostructures. This section presents the fundamental design principles by going through all the different phases in the process. Finally, the concept of design software is described.

3.1 Process

As discussed earlier, nucleic acid strands can be assembled into a desired conformation *via* rational design. With the help of a CAD (Computer-Aided Design) software, designing the target structure is rather straightforward. The overall process obeys the following order: first the desired target structure is specified, then the arrangement of nucleic acid strands in the target structure is defined, and the last step is to specify the (base) sequences of each nucleic acid strand. Furthermore, it must be taken into account whether the structure is static or dynamic, rigid or flexible, is the assembly symmetric or non-symmetric, is the structure periodic, in how many dimensions the scaffold routing should take place and is the structure infinite or not. Based on the previous requirements, building blocks for the desired structure need to be selected. Some of the possible building blocks are illustrated in Figure 3.1. [27] The structural design step varies depending on the type of the target object. The different types of structures are discussed in the next section.

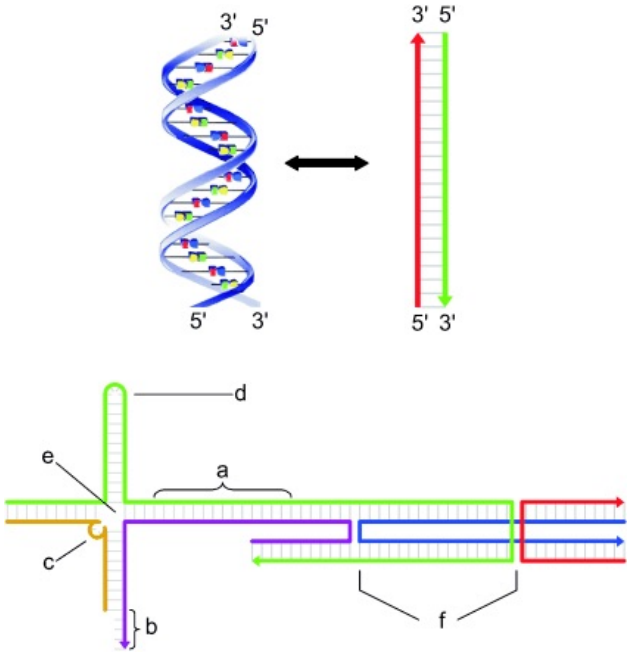

Figure 3.1: DNA double helix and different kind of motifs: (a) double-helical region, (b) sticky end, (c) bulge loop, (d) hairpin loop, (e) junction, and (f) crossovers [27]. Top: DNA double helix is formed by two anti-parallel and sequence-complementary oligonucleotide strands.

3.1.1 Structural design

The first step in the design process is to determine the structure of a desired shape. For example, the following structures can be utilized:

- Tile-based structures
- Dynamic assembly
- Folding structures (DNA origami)

In the tile-based self-assembly the end result conformation consists of a repeating pattern. The key concept is to use branches or tiles with sticky-ends as building blocks. For example, in the original idea of Nadrian Seeman, the sticky-end associations of junctions are connected to create periodic 2D and 3D lattices. Figure 2.4(a) shows the idea of four-arm branched junction. [59]

In dynamic assembly the devices are reconfigurable and possibly autonomous. Moreover, the end-states of such devices are at non-equilibrium. The desired dynamic functionality can be achieved, for example, by using

hairpin structures [105], which then assemble into the final configuration through DNA strand displacement reactions [107]. In such process two strands hybridize to each other by displacing one or more previously hybridized strands. In general, the conformation of a complex can be changed *via* an external stimulus, such as temperature, pH or salt concentration. However, DNA strand displacement mechanism makes it possible to address devices in a sequence-specific manner. In addition, dynamic assembly/disassembly between the designed DNA nanostructures can be achieved through shape-complementary domains [31]. The main focus in this thesis is on the static structures and on their design process, and therefore, dynamic assembly will not be further discussed.

When designing nanostructures using the folding approach, a long single-stranded DNA is directed into a desired shape as mentioned earlier in this thesis. There are two different approaches to achieve the desired result: the long strand can fold with itself or the long strand can be folded by using a specific set of short 'staple strands'. The latter method is called a 'DNA origami', and designing such structures requires the fundamental steps described below.

First, a geometric model of a desired structure is approximated by building it with the help of CAD software or by hand. In Figure 3.2(a) the red outline illustrates the desired shape, which is then filled from top to bottom using parallel double helical domains (depicted as gray cylinders in the figure). Moreover, in Figure 3.2(a) there are small blue crosses that indicate crossover positions, *i.e.* the locations where the strands running along one helix switch to an adjacent helix and continue there.

In the second step, as shown in Figure 3.2(b), a long scaffold strand is folded back and forth in a raster-fill pattern, and it constructs one of the two strands in every helix. When crossing from one helix to another, an additional set of crossovers (scaffold crossovers), illustrated as red crosses, are created. The scaffold can form crossovers only at points where the distance between adjacent helices is the shortest. By performing this step, a 'seam', where the path does not cross, is formed in the middle in this exemplary structure. After these two steps, the set of 'staple strands' is designed.

In Figure 3.2(c) the colored DNA strands conform periodic crossovers. The strands run in an antiparallel way with respect to the scaffold and therefore provide Watson-Crick complements. The helices will likely bend a little bit in the structure so that only a single phosphate from each backbone occurs in the gap (no bases are 'spent' in the staple or scaffold 'hinge'), and the algorithm uses only an integer number of bases between periodic crossovers. For example, the helical rise of B-DNA is $\sim$10.5 base pairs (bp) per turn, but the used design program might instead use for example 16 bp for 1.5 turns

(10.667 base pairs (bp) per turn), similarly as in the case of the examplary structure shown here. Therefore, the caused strain could be balanced by changing the position (usually by a single bp) of crossovers.

Finally, in order to achieve higher binding energy and more stable structures, selected adjacent staple pairs could be merged resulting in longer staples (this should be taken into account in every origami design). In practice, the staples are typically 15-60 bases long (easy and not too expensive to synthesize), and the staple breaks are designed in such a way that the structure could be as stable as possible [69]. Furthermore, a seam can be strengthened by 'bridging' it with the staples (merging the staples across the seam). Figure 3.2(e) shows the finished design after all the arrangements and merges. [80] As a final note to this section, it should be mentioned that the staples are rarely merged in the design process, and instead, the design software (such as caDNAno, see the next chapters) creates as long staple strands as possible, so that the designer has to break them manually or by using automated cutting tools.

3.1.2 Sequence design

After the target secondary structure has been designed as discussed above, the actual nucleotide sequence, that will adopt the intended shape, must be determined. In this step a specific sequence of nucleic acid bases are assigned to the designed strands. However, there are several possibilities to select the sequences of nucleic acid strands, but in order to avoid undesired additional interactions and to achieve the desired structure, there are several different criteria that could be taken into account. In the article conducted by Dirks *et al.* [18] they studied the following criteria for choosing sequences:

- Random
- Energy minimization
- Minimum free energy satisfaction
- Sequence symmetry minimization
- Probability
- Average incorrect nucleotides

In *random* approach sequences are selected randomly, but satisfying complementary requirements. *Energy minimization* means that sequences attaining a low energy on the target structure are selected. In *minimum free energy satisfaction* sequences are chosen in a way that ensures that the target structure is the lowest energy structure. This approach may not adopt the target

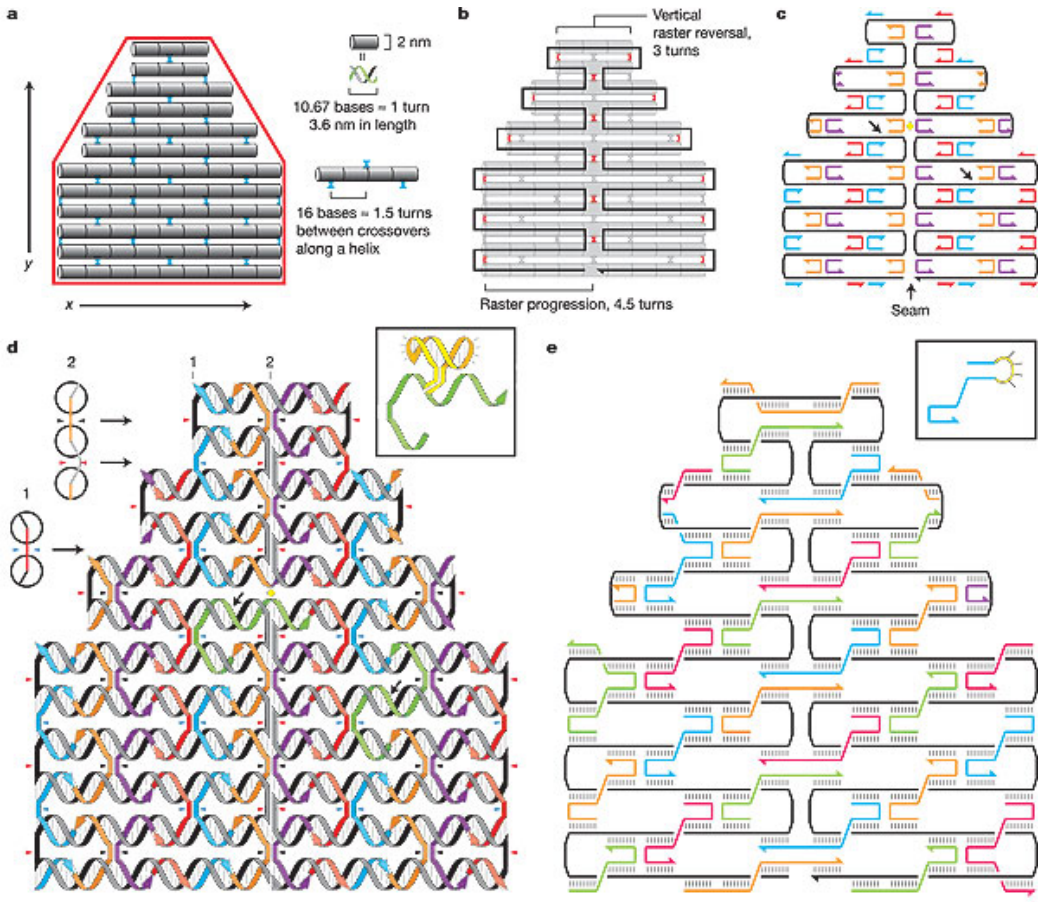

Figure 3.2: DNA origami design. (a) The target shape is depicted with the red line and it is approximated by the gray dsDNA cylinders. (b) Black scaffold is routed to run through every helix. (c) Staple strands (colored strands) bind to the pre-defined locations of the scaffold. (d) Strands drawn as helices (similar to (c)). (e) Similar but strengthened design can be achieved by rearranging the staples along the seam [80].

fold, although the sequence produces correct minimum energy structure. *Sequence symmetry minimization* means that sequences are selected so that there exists no repetitive subsequences. The purpose is to ensure that no undesired hybridization should occur in the process. *Probability* means that sequences are selected to maximize the probability of achieving the target structure and average incorrect nucleotides criteria simply implies that the average number of incorrect nucleotides is minimized.

Main purpose of nucleic acid design is to ensure that the complexes are not too strained and that the target structure is thermodynamically propitious. The selected criteria are ambiguous and different applications may have somewhat different design criteria. Nowadays there exist several publicly available design programs to generate the sequences. [12]

It should be noted here that different DNA origami nanostructures are usually designed using the same scaffold strand (M13mp18). This means that in that sense there is not a fundamental sequence selection criteria. However, in some cases, it would be useful to take into account the differences (melting temperature, secondary structure *etc.*) of the double-stranded domains, and accordingly select the crossovers and lengths of the staple strands to maximize the stability of the structure as a whole.

It should be noted here that programs such as Sequin [90], NUPACK [106] and mfold [112] are currently used widely as important tools in analyzing DNA sequences and base pairing properties. Sequin was used for example in designing the strands for the first 3D DNA crystals [110].

3.1.3 Folding and purification

This part of the process is fully experimental and typically carried out in a biochemistry laboratory. Therefore it is not thoroughly studied in this thesis, but since it is an important part of the procedure, the main principles are briefly discussed here. After the target structure and the sequences have been designed, the commercially available DNA strands are mixed together in an aqueous solution. Usually some cations (typically sodium or magnesium [69]) are added to the mixture in order to screen the repulsive force between the negatively charged strands. In a standard procedure, the structures are formed by annealing: the strand mixture is heated up to 65–90 °C (melting the strands) [21, 80] and slowly cooled down to room temperature (assembly of the strands). However, it has been observed that the structures can be rapidly formed using constant temperatures (folding temperatures are specific for each design) [72, 95].

Once the structures have been formed and their folding quality has been verified by agarose gel electrophoresis, AFM or transmission electron microscopy (TEM) [13], they can be purified by removing the excess strands and possible misfolded structures. This can be realized by utilizing, for example, spin-filtering [52], polyethylene glycol (PEG)-purification [96] or rate-zonal centrifugation [60]. The purification methods allow an effortless buffer exchange (the folding buffer changed into the buffer of choice), and therefore, after the purification, the customized DNA nanostructures are ready to use for a great variety of applications.

3.2 Computer-aided design software

Simply put, computer-aided design means any type of activity involving the use of computer to develop, analyze or modify an engineering design. Computer-aided design (CAD) software assists in the creation, modification and optimization of a design. Usually CAD systems are based on interactive computer graphics, where the computer displays the data in the form of pictures, symbols or basic geometric elements.

There are several advantages of using CAD software. First of all, the productivity of the designer can be increased. The designer is able to visualize the product and this often shortens the time of a design process. Of course, much depends on the complexity of the design or on the fact how many repetitive components there exist. Secondly, the quality of the design is usually improved and the number of design errors are reduced. This is achieved, for example, by doing calculations and checks by the program. Thirdly, the software often offers design analysis, which helps to optimize the design without performing feedback sessions or separate analysis procedures. The designer can simply perform the analysis while working on a CAD system, and this way, the process becomes more interactive. Finally, the design is more understandable, especially when designing 3D models. 3D views make it easier to comprehend the design and this enables direct visualization instead of imagining the actual shape from the 2D objects. [83]

The next chapter is devoted to comparison of the computer-aided design software for creating versatile nucleic acid nanostructures.

Chapter 4

Comparison of design software

While the field itself is growing rapidly, the software for designing needed nanostructures have seemed to be somewhat dragging behind for a long time; most of the applications have been partially outdated and most of them have lacked or are still lacking comprehensive operating instructions. Moreover, the common software require specific version of each operating system or other dependent software to function properly. In some cases the software could have been developed merely for some specific purpose. In this chapter the criteria for the comparison of design software is set and the most recent and relevant available programs are tested against those criteria. Additionally, each of the application is briefly introduced, and the advantages and disadvantages of each program are assessed. Using these software, designs of simple exemplary structures are also demonstrated. Finally, the findings and observations are discussed, and some guidelines are given for the reader to choose the most suitable software for any desired shape or application.

4.1 Comparison criteria

Since there is no fundamental approach for comparing software, the comparison criteria defined in this work are based on what is thought to be the most suitable from a perspective of a regular user. Here, a regular user means a person who has no programming background, but possesses basic computer skills. The criteria are:

- Availability on different platforms
- Usability, user-friendliness and intuitiveness
- Documentation
- Visualization and graphics

- Support for different kind of objects: 2D, 3D, possible twists and bends, wireframe structures
- Level of automation
- Export formats *e.g.* for the DNA synthesis and the simulation software

Availability is checked between Microsoft Windows, Linux and Apple OS X, and the compatibility with different versions is taken into account as well. Usability, user-friendliness and intuitiveness is estimated by designing simple and similar nanostructures with each software. Moreover, user-friendliness is evaluated by assessing the way a program informs user about unexpected events or forbidden operations. Documentation criteria refers to how thoroughly a program is documented and how easy it is to start using the program by going through the available documentation. Visualization and graphics are simply evaluated from the usability and user-friendliness point of view; how intuitively the structure is displayed to the user and how the actual model is depicted (2D, 3D *etc.*).

The objects supported by a software are essential: they have effect on the aspect how complex structures can be designed. For example, in order to twist or bend an object, the design software should allow insertions or deletions of base pairs [17]. In general, a wireframe structure is a model where the shape is presented as lines and points. From the point of view of DNA nanotechnology, the wireframe structure means that the target pattern is treated as a planar graph, where intersections between edges are vertices and the edges do not cross each other. As illustrated in Figure 4.1, the first step is to create an arbitrary pattern of line segments and vertices. Next, all the connections are converted into double lines, which are looped and bridged to create a single continuous line. This line is the basis for one single-stranded scaffold, so that the scaffold can travel through all the vertices. Crossovers are placed in such a way that the scaffolds of two adjacent lines are always antiparallel. Finally, the complementary staple strands are added to the design. [108]

A level of automation refers to the fact how little human input is required in the whole process. The feature 'export formats' means the file types that can be exported from the program as an output. For example, it tells if the structures can be easily verified with simulation software or if the structures can be examined and viewed as atomic models. Furthermore, the way DNA sequences are exported is taken into consideration.

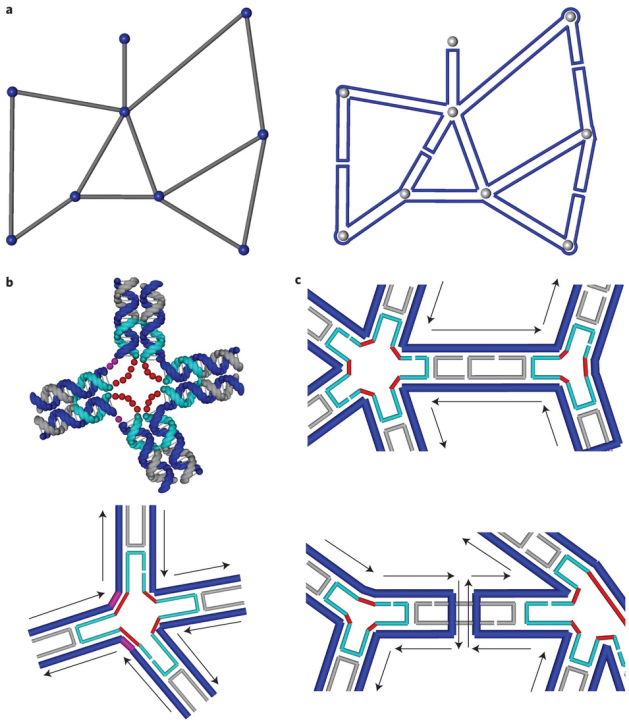

Figure 4.1: Wireframe DNA origami structure [108]. (a) Left: an arbitrary wireframe composed of line segments. Right: all the line segments doubled and bridged to route one continuous scaffold. (b) Top: DNA helical model of a junction at the vertex (blue dot in (a)). Bottom: a line model of the same junction. (c) Staple strands are added to two different edge types. Dark blue lines represent the scaffold, whereas gray and cyan lines depict the staple strands.

4.2 Tested programs

In this section the earliest software as well as the more recent and more sophisticated programs are reviewed. The software found suitable for the comparison are tested (exemplary designs created) and reported in more detail.

4.2.1 Earliest software: GIDEON and SARSE

The first design software for DNA nanostructures was developed by Ned Seeman and colleagues. It was called GIDEON [11], and it was published in 2006. However, soon after its release it was forgotten by many researchers,

since the other techniques – such as DNA origami – emerged rapidly. Therefore, GIDEON program is not further discussed here, but nevertheless it is mentioned due to its historical significance.

After the DNA origami was published in 2006, the researchers quickly adopted this new and revolutionary method, and subsequently, they started to develop user-friendly design software for DNA origami nanostructures. This effort yielded first the software SARSE and soon after that even more advanced tools (see below).

SARSE [3] was the earliest publicly accessible tool for designing DNA origami structures. It is available for Mac and Linux and it requires Java Virtual Machine to run. Although the software is rather old (2008), it runs without troubles with the latest Java version (1.8.0) on Mac OS X El Capitan. The documentation and tutorials cannot be found from the website nor from the software itself, and therefore SARSE is quite difficult to learn.

SARSE contains some special features: bitmap reader, automatic 'origami folding system', 3D generator and oligotracker (Fig. 4.2). The bitmap reader should be able to transfer images into a shape, and the imported shape could then be further modified in the editor. However, when the software was tested in this work, the bitmap reader did not function properly, and it just produced errors.

According to the original article [3], the origami folder is able to automatically find a folding path through the shape and add staple strands with crossovers as well. 3D generator generates 3D atomic model of the structure which can be viewed, for example, with PyMol (molecular visualization system) [15] or QuteMol [97], and finally oligotracker is used to insert a given sequence of nucleotides in the scaffold and export an output list of staple strands and their sequences.

As mentioned above, SARSE had already some interesting features and it was designed specifically for DNA origami approach, but the lack of documentation combined with a relatively ascetic user interface (see Fig. 4.2) makes it difficult to use nowadays. It seems the development and maintenance of this software have been brought to a halt, and the users in the field of structural DNA nanotechnology have now begun to use more recent software such as caDNAno (see the next section).

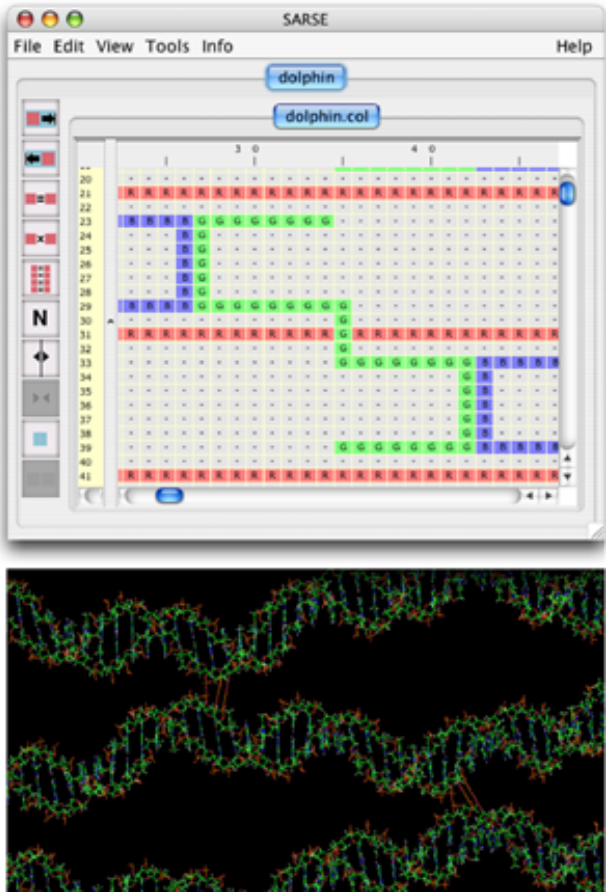

Figure 4.2: Top: SARSE main window. Bottom: 3D generated atomic model of the design [3].

4.2.2 caDNAno and CanDo

caDNAno [22] and CanDo [13, 46, 53] are widely used together for designing DNA origami nanostructures and computationally predicting the created shape. caDNAno was originally developed in William Shih's laboratory[1] at the Dana Farber Cancer Institute[2] at Harvard University, but the development has since been moved to Wyss Institute for Biologically Inspired Engineering[3], also located at Harvard University. Current caDNAno team consist of Nick Conway and Shawn Douglas, and the first public version of

[1]http://research4.dfci.harvard.edu/shih/SHIH_LAB/index.html
[2]http://www.dana-farber.org/
[3]http://wyss.harvard.edu/

the software was published in 2009. Updates are applied regularly, and the newest update is from November 2014. Windows and Mac OS X are supported, although the latest version is not compatible with OS X El Capitan. Newer version (2.5)[4] of caDNAno is under development, and that version should fix the compatibility issues with OS X once it is released. However, the release date of the update is not available when writing this thesis.

In the older versions of caDNAno (such as v0.2), there was a direct visualization tool available. Currently, caDNAno should contain a 3D interface provided by Autodesk Maya, but the feature did not work when tested here and therefore it was not taken into account when assessing usability and visualization. However, 2D cross-sectional view is often enough to design simple structures. One of the great features is that caDNAno is open architecture and allows plugins to be created in order to expand the basic functionality. Unfortunately, running the program from the source code is difficult, because required third party libraries are old and unpleasant to install. Therefore creating plugins is quite demanding at the moment, but the upcoming release should fix the problem.

The core purpose of caDNAno is to simplify and speed up the design process of DNA nanostructures. The usual design process in caDNAno is divided into four steps. First, the desired shape is approximated by inserting helices into the predefined lattice (either honeycomb or square lattice geometry) (Fig. 4.3 left panel) and then the scaffold path that passes between neighboring helices is constructed (Fig. 4.3 right panel). The scaffold path has to include antiparallel crossovers to connect neighboring helices. The goal is to construct a scaffold path in a rasterized manner meaning that the path is continuous from the beginning to the end. Once the scaffold is completed, staple strands (Fig. 4.3 colored lines on the right panel) should be inserted. By default caDNAno inserts all allowed staple crossovers, except the ones that are five base pairs away from a scaffold crossover between the same two helices. Next, the staple strands are splitted into shorter segments (18 to 49 bases long) and finally, the desired DNA or RNA sequence is inserted into scaffold and the complementary staple strand sequences are generated automatically. [22]

The lattice grid is divided by 7 base pair segments for the honeycomb lattice and in 8 base pair segments for the square lattice geometries. The 8 base pair rule was used already in Rothemund's seminal work [80], where the 2D structures were actually single-layer structures in the square lattice geometry (in caDNAno view). By following the helical pitch in a 2D structure, it can be noticed that after two full segments the helix has turned $\sim$1.5 rounds al-

[4]https://github.com/cadnano/cadnano2.5

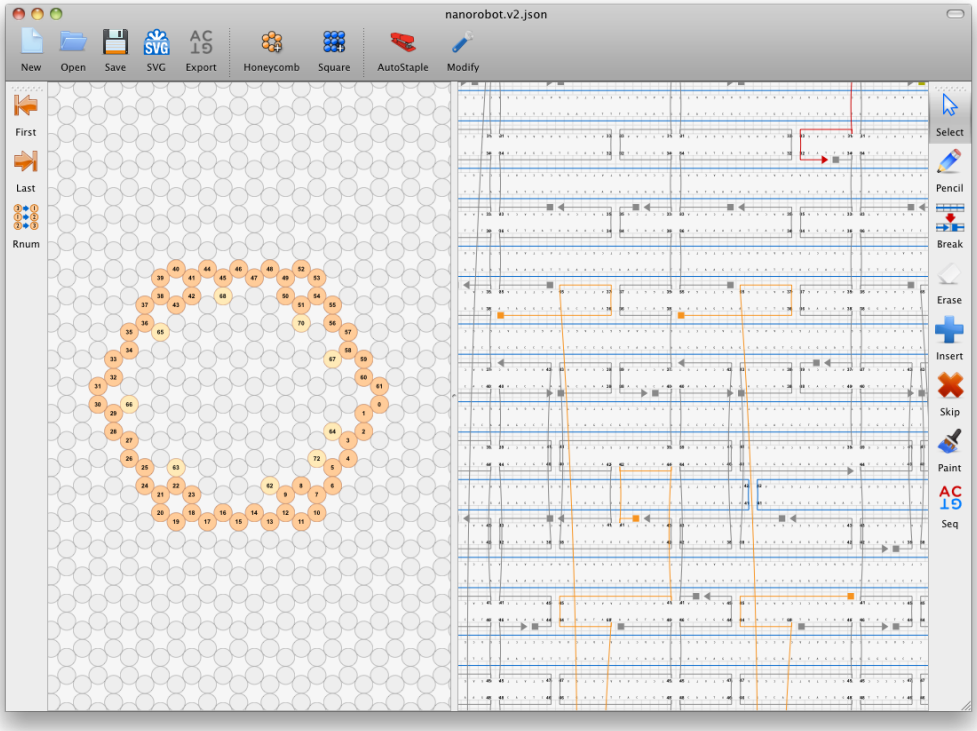

Figure 4.3: caDNAno user interface [22]. On the left-hand side: a honeycomb lattice (square lattice also supported, see other figures below), where the desired helices can be selected. On the right-hand side: unrolled 2D illustration of the scaffold routing and the staple strands.

lowing a next crossover position to a neighboring helix. However, this yields actually 10.67 bp per helical turn, whereas the same value for the dsDNA is ∼10.5 bp. Therefore, without compensating this by extra base pairs in the design, the designed structure will naturally adopt slightly right-handed twist. However, in the honeycomb lattice, the 7 base pair rule per segment yields structures without twist as explained in Figure 4.4. After each 7 bp rise, the helix has turned 240° thus pointing toward the neighboring helix. After 3 full segments, the helix is again pointing to its original direction. However, one can tune the curvature, twist or bend of the structure by deleting or inserting base pairs [17]. Deletion of base pairs from the segments leads to a left-handed torque (less than 240° turn) and inserting base pairs induces a right-handed torque (more than 240° turn). This feature can be fully controlled, since the twisting or bending can be simulated by CanDo software

(as can be seen when simple brick-like structures are simulated below).

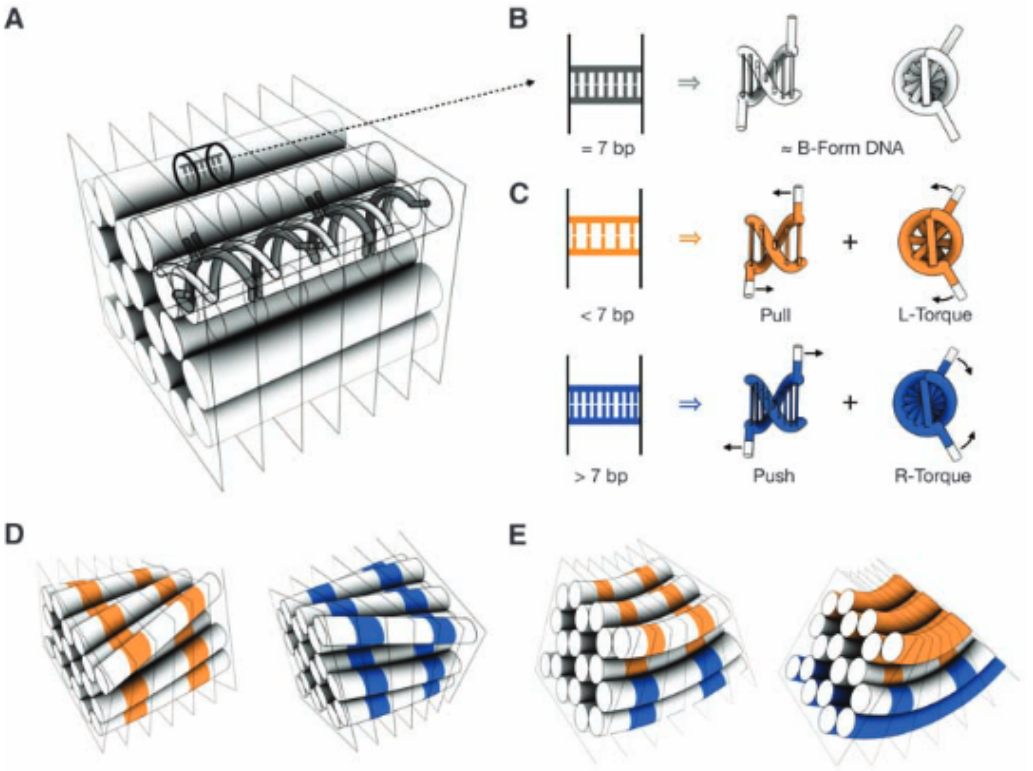

Figure 4.4: Honeycomb lattice geometry in caDNAno [17]. (a) The lattice is divided into 7 bp segments. (b) Choosing exactly 7 bp for each segment yields to the natural B-form DNA. (c) Less than 7 bp per segment results in a left-handed torque, and more than 7 bp per segment induces a right-handed torque. (d) Left- and right-handed twists of the honeycomb bundles. (e) Bending of the structural elements can be achieved and tuned by combining less than 7 bp and more than 7 bp segments.

One of the advantages in caDNAno is that it uses visual guides and predetermined rules to aid with the process and ensure the correctness of the design by showing the possible crossover positions between neighboring strands. Crossovers can also be forced between any two strands. Furthermore, caDNAno contains a variety of tools to make the design task easier. For example, the scaffold can be automatically rasterized and crossovers inserted by click-and-dragging with the mouse on the helix. In order to add staple strands as conveniently as possible, the auto-staple tool can be used. It generates a default set of staples with maximum number of possible crossovers. Staples

that are not between the preferred length (18-49 bases) are highlighted. All staple strands can be further modified with the 'breakpoint tool', which splits the paths into shorter lengths defined by the user. As mentioned already, caDNAno should also provide a 3D view of the design (powered by Autodesk Maya), but this feature was not testable here due to incorrect version of the Maya.

The documentation in caDNAno is not thorough. Documentation section on the website contains only links to YouTube, but by examining those video clips, it is rather effortless to start creating new nanostructures. In these clips the geometric explanation of possible crossover positions is given, caDNAno user interface is explained, an example structure is designed with it, automatic scaffold rasterization is explained, and selection-based editing is demonstrated. Overall, caDNAno is intuitive enough to use, and important visual clues, such as marked crossover positions, make it user-friendly (user knows immediately where the crossovers can naturally take place). On the other hand, the user is not well-informed about different operations or on error situations. For example, if the autobreak operation is not possible, there are no hints about what was missing or what has simply gone wrong in the process. In addition, the program crashes quite often on OS X, so it is important to save the project frequently.

In general, with caDNAno it is possible to design 2D and 3D objects. Twists and bends are not directly supported, but it is possible to skip and insert base pairs at desired locations. As explained previously, skipping or inserting bases results in the twisting or bending of the structure [17]. Unfortunately, there is no direct visualization for this feature and for that reason designing such structures might be tedious. For such features, a powerful simulation software is needed.

As mentioned above, CanDo is such a tool for predicting appearance, mechanical fluctuations and flexibility, as well as twists and bends of the particular shape in the solution. The software was developed by Hendrik Dietz's (Technische Universität München) and Mark Bathe's groups (Massachusetts Institute of Technology, MIT) [13, 46], and it is currently maintained by the Laboratory for Computational Biology and Biophysics at MIT[5]. The idea is to offer computational feedback (finite-element based rigid-beam model) in order to reduce financial costs and time to design nanostructures successfully (try-and-error process). For example, it can be used to verify wrongly designed or otherwise undesired structural features already before synthesis. CanDo directly supports caDNAno project files (.json), and is therefore often used in conjunction with the caDNAno design procedure. Figure 4.5 displays

[5]http://lcbb.mit.edu/

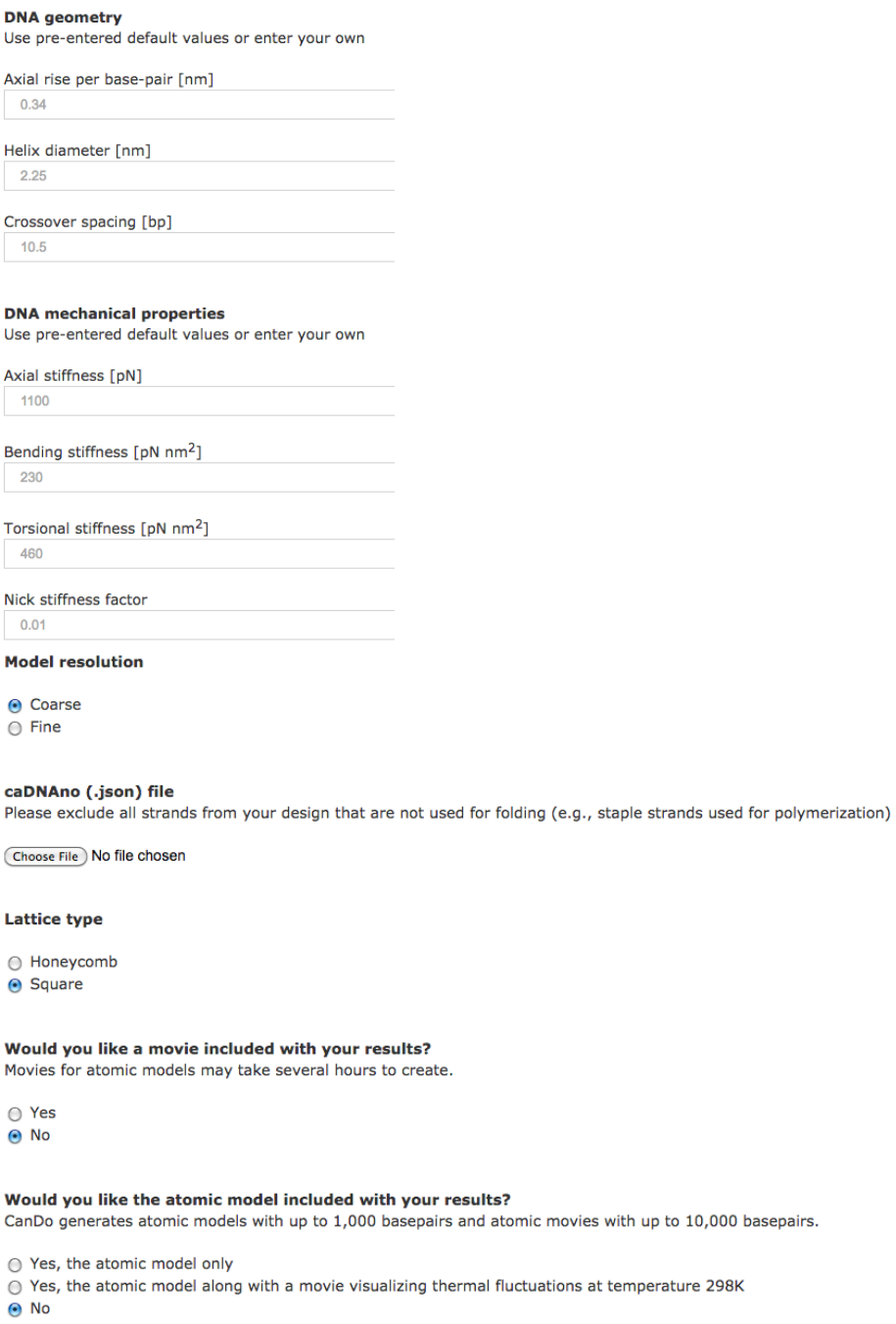

Figure 4.5: Screenshot of CanDo user interface.

CanDo user interface and the available parameters that can be used. The default parameters are sufficient for most of the cases, and for example an

atomic model (requires also a caDNAno sequence file as .csv) or a movie of the simulation result can be easily produced.

Below, two exemplary brick-like structures are presented. Bot of them are designed using caDNAno and simulated with CanDo. 60-helix bundle (60HB) design (shown in Figure 4.6) has been taken from the article by Linko *et al.* [67] and 64-helix bundle (64HB) (shown in Figure 4.7) has been designed specifically for this thesis in order to test design properties in the square lattice geometry, autobreak command in caDNAno software and simulation of suca a structure using CanDo.

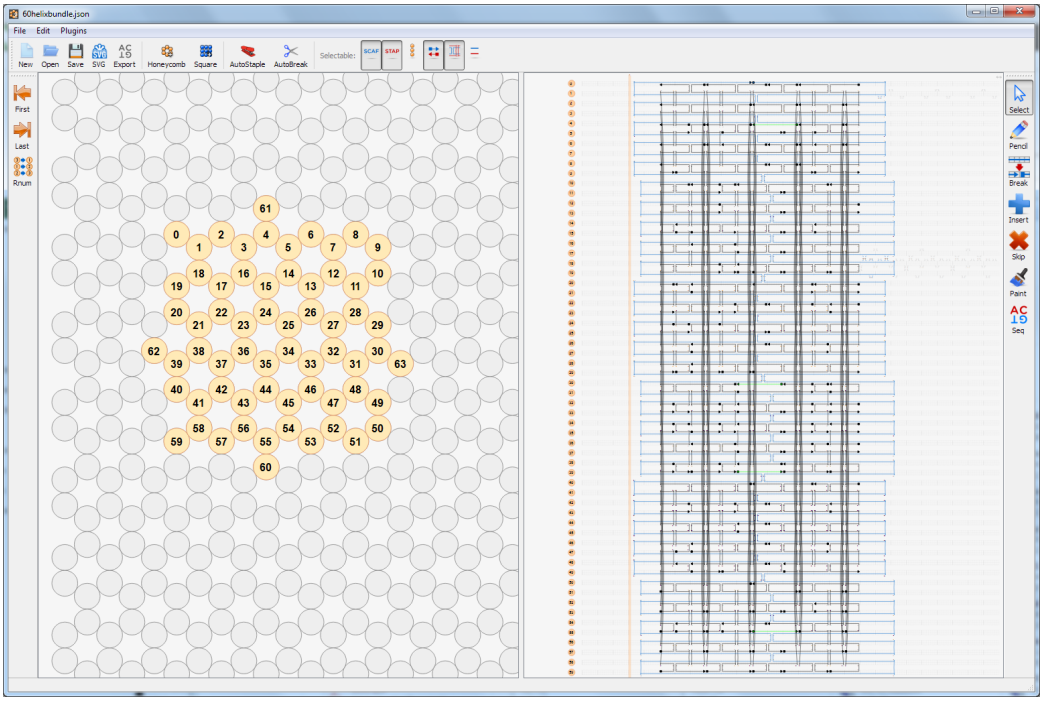

Figure 4.6: 60-helix bundle structure design in caDNAno [67]. The actual 60HB structure is formed by the helices 0-59, and the helices 60-63 are only added to indicate the locations of the possible side-modifications.

Both structures are designed in layer-by-layer fashion, and the CanDo simulated deformed solution shapes of the designs are shown in Figure 4.8 (.bild files). It is noteworthy that the sidestrands are omitted in both designs in order to avoid blunt-end stacking, and CanDo only depicts full dsDNA segments (scaffold loops not visible). It can be observed from the CanDo simulations that although the brick-like shapes are quite rigid, the square lattice packing [43] of 64HB induces slight right-handed twist in the structure

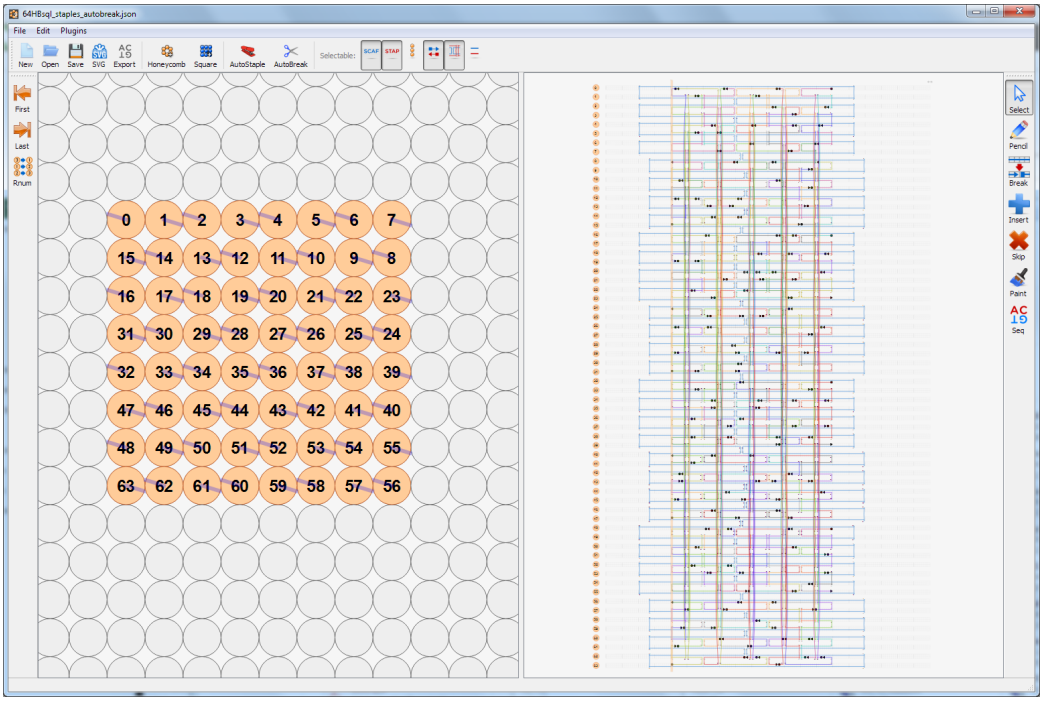

Figure 4.7: 64-helix bundle structure design in caDNAno.

due to the non-ideal periodic rules described above, whereas the honeycomb lattice packing yields a straight structure. This pair of exemplary structures describes well why the software CanDo is really useful for caDNAno users. CanDo can be efficiently used to resolve and verify the actual shape of the origami structure before the synthesis. For instance, one can simulate how bent or twisted the structure is, and try to iterate the design in order to get a desired result.

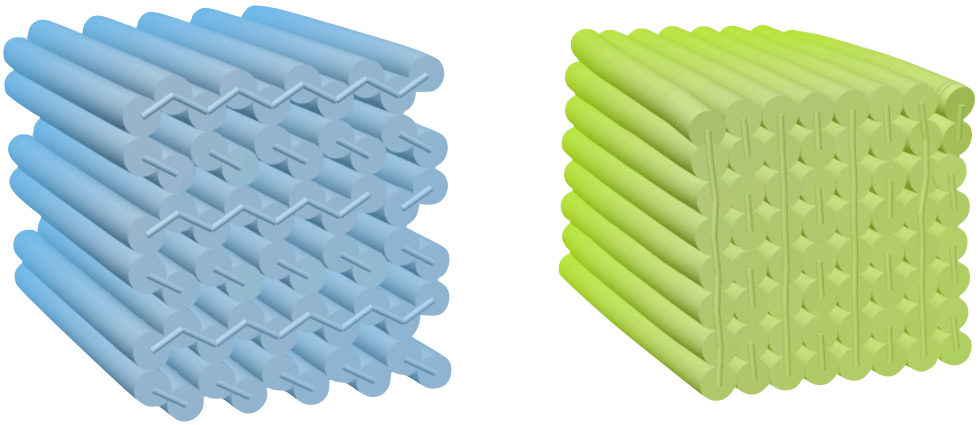

Figure 4.8: CanDo-simulated deformed solution shapes of 60-helix bundle (60HB, blue) designed in honeycomb lattice [67] and 64-helix bundle (64HB, green) in square lattice. Each cylinder represents a dsDNA molecule.

4.2.3 Tiamat

Tiamat [103] has been developed in 2009 by the researchers of Yan Lab at the Arizona State University[6]. It was developed to overcome the fundamental limitations of GIDEON and Sequin, and it has specifically been designed for more complex DNA nanostructures. GIDEON and Sequin could not handle large structures properly and the visualization with them was inadequate.

Tiamat uses lattice-free approach as opposed to caDNAno. Very recently introduced wireframe technique DAEDALUS [98] (see next Sections) and its lattice-free simulation/visualization tool [74] is partially based on the Tiamat software features (same simulation tool can be used to simulate Tiamat design files (.dna)). Tiamat is based on combining double-stranded DNA elements together by hand, and therefore it can be used to create simple geometries, such as different tiles (for example a DX tile, which serves as an edge of all DAEDALUS shapes) or a tetrahedron (Fig. 4.9). However, at present, the software can be considered somewhat outdated, since there are more versatile lattice-free techniques available. Moreover, the available Tiamat tutorials are insufficient, making the design process relatively challenging for a newcomer. For these reasons, Tiamat has not been used here to design an actual exemplary structure, but nevertheless its basic features have still been listed in the Table 4.1 along with caDNAno (see the previous sec-

[6]http://yanlab.asu.edu/

tion) and the most recent software vHelix and DAEDALUS (both described in the next sections).

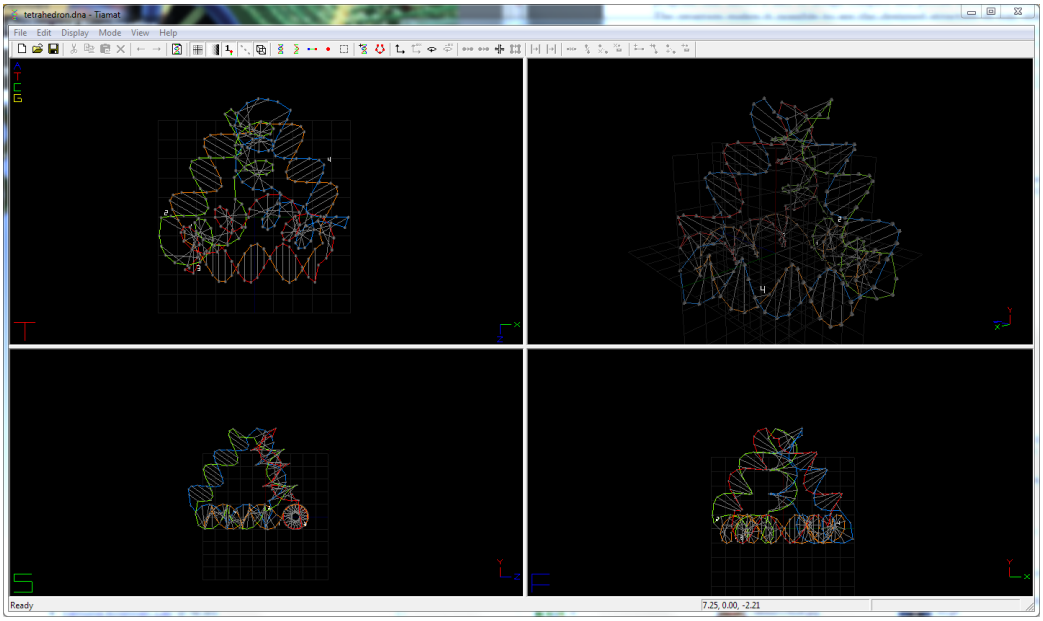

Figure 4.9: Screenshot of a tetrahedron structure in Tiamat user interface. The interface is divided into top, side, front and perspective views.

4.2.4 vHelix

vHelix [10] has been developed in the lab of Björn Högberg[7] at Karolinska Institutet[8]. vHelix aims to overcome the limitations that exist in previous design software, such as the complexity of the overall design process. The core idea is to automate the scaffolding process and thus eliminate extra manual work by hand, and therefore allow a user to create more complex structures. vHelix uses routeing algorithm based on graph theory and a relaxation simulation that creates scaffold strands using a given target structure (3D model) as an input. vHelix works for structures that enclose a volume, which can be inflated into a ball (spherical topologies) and the fundamental idea is to replace all the edges of the target mesh with single DNA double helices. Furthermore, the constructed scaffold strand should only traverse each of the edges once, thus forming the Chinese postman problem: there is

[7]http://www.hogberglab.net/
[8]http://ki.se/

a closed 'walk' covering every edge only once. In addition, there are three principles that need to be satisfied. First, meshes should be allowed to be triangulated in order to increase structural rigidity. Second, each edge of the target mesh should be represented by just a single double helix to keep the amount of used DNA as low as possible hereby enabling construction of large structures, and finally, the scaffold should not cross itself at the vertices minimizing topological and kinetic traps during the folding procedure.

vHelix runs as a plugin in Autodesk Maya versions 2011, 2012, 2013, 2014 and 2016. Other versions as well as 32-bit versions are supported only by manually compiling the plugin from the source code. Supported platforms are Windows, Mac OS X and Linux. One of the disadvantages of requiring Maya is the high cost of the software, although students can get it for free for three years (at Aalto University). Overall, the installation process is straightforward (at least on Windows) with the instructions found from the website; once Maya is installed, the vHelix plugin file can be simply copied to Maya's plugins directory, and then it is enabled via the plugin manager. Additionally, the scaffold routing algorithm and the spring relaxation are found from a separate software called BSCOR, which should be downloaded and run on a local machine. The BSCOR binaries are only for Windows, but an online version[9] of the software is also available for non-Windows users.

As Tiamat, vHelix is also a lattice-free software, and the structures constructed with vHelix are based on the wireframe approach. Figure 4.10 depicts the design and automated workflow of 3D origami meshes. The first step (Fig. 4.10(a)) is to construct a polyhedral (volume inflatable to a spherical topology) 3D mesh of the target nanoscale geometry using some of the available 3D modeling software. Next (Fig. 4.10(b-e)), the long scaffold strand, that traverses through all the edges of the mesh, is generated. Then (Fig. 4.10(f-i)), the least-strained DNA helix arrangement is determined and finally (Fig. 4.10(j)) the generated staple strands imported in Maya could be fine tuned by a user. For example, staple-strand breakpoints can be modified. After adjustments vHelix automatically generates the staple-strand sequences that can be ordered. The final structure is comprised of the maximum number of single duplex edges (scaffold routing algorithm maximizes this) so that it minimizes the amount of DNA used in the design.

As an example, a simple cube is designed in this thesis in order to demonstrate the process concretely. The first step is to create the target shape in Maya and after using the Maya's cube creation tool, the initial design along with the vHelix menu is shown in Figure 4.11. Next, the model is exported in STL-format (as instructed in tutorial videos) to hard drive and converted

[9]http://www.vhelix.net//index.php?cID=128

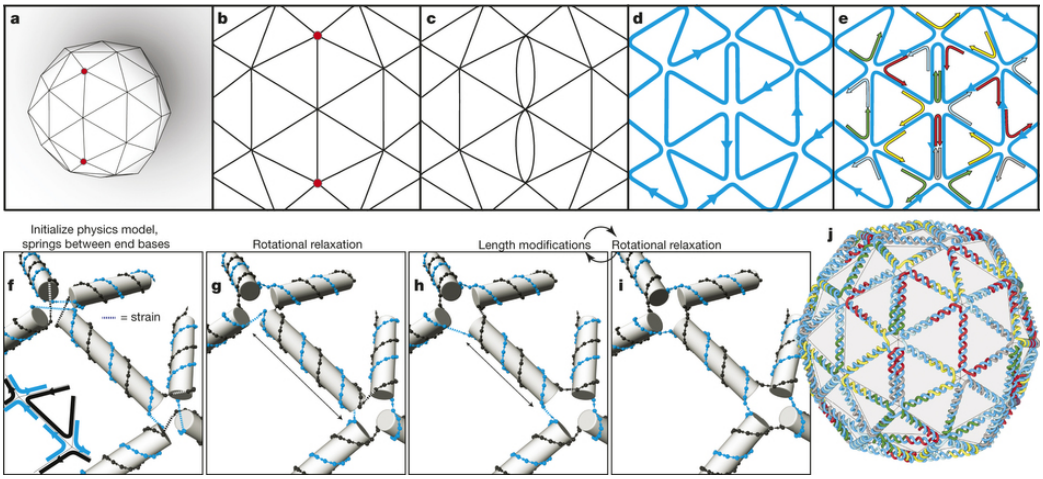

Figure 4.10: vHelix design workflow [10]. (a) Input 3D model in which gray lines represent edges. (b) The algorithm pairs odd-degree vertices. (c) Extra helices (helper edges around certain positions) are introduced in order to satisfy Eulerian circuit (trail starting and ending on the same vertex). (d) A-trails algorithm generates the scaffold. (e) Staple strands are added. (f)-(i) Strain is relaxed and evenly distributed using a physics model. (j) The final design.

into PLY-format. This step is required, because Maya does not support PLY-format directly, and on the other hand, the scaffold routing software only supports the PLY models. By using meshconv[10], the file can be converted from the command prompt by using the following command:

```
meshconv cube.stl -c ply -ascii
```

After that, the routing and physical relaxation software can be run in order to generate the scaffold strand. This is carried out by running the command below. Number 1.0 is the scaling value.

```
bscor.bat cube.ply 1.0
```

The software creates .rpoly-file, which can be imported back to Maya. After that, the final scaffold along with the staple strands is illustrated in Figure 4.12. Finally, the sequence for the scaffold strand can be applied by first selecting one of the scaffold bases and then applying the desired sequence to it (*e.g.* M13mp18) from vHelix menu. Afterwards the strand sequences can be exported as comma separated value list (CSV) for further use.

[10]http://www.patrickmin.com/meshconv/

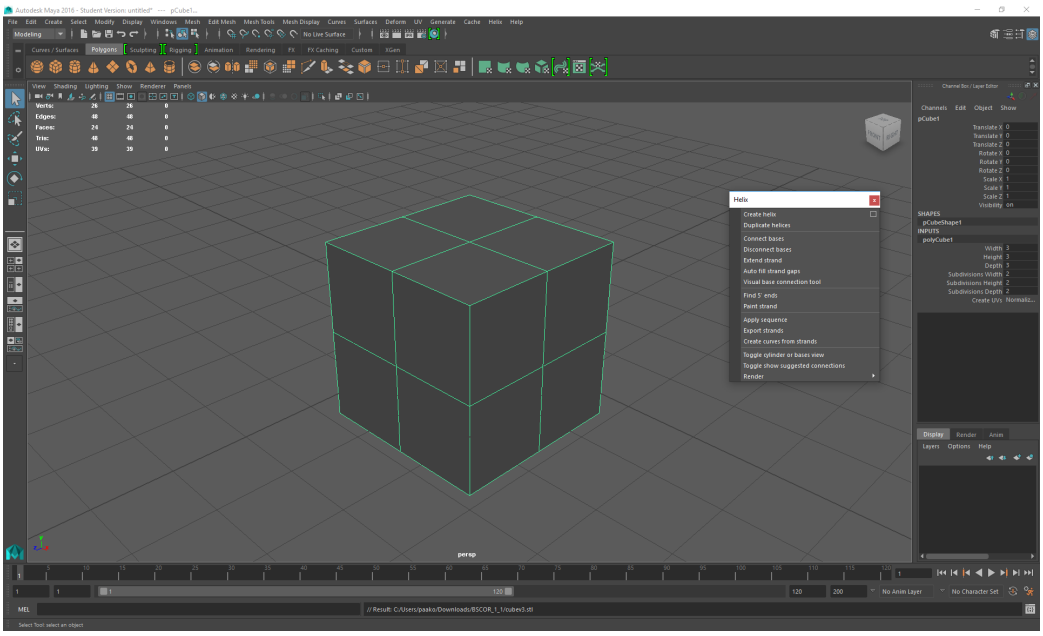

Figure 4.11: Initial 3D model of the target shape.

Because the whole process is almost fully automated, more effort is required in mastering the modeling software, such as Autodesk Maya, than in using vHelix itself. Once a 3D model is constructed, vHelix is simple to use. vHelix offers easy to follow Youtube video tutorials on their website and after completing the tutorials, using the software should be rather trivial. vHelix offers a single menu where all the functionality is located, thus making it intuitive to use. As said, Autodesk Maya itself requires more effort to master, but most of the necessary features are demonstrated in vHelix tutorial videos.

Very recently, the developers of vHelix (led by Björn Högberg's group) introduced an extension to the original software. This extension allows one to create 2D origamis using flat-sheet meshes [9]. The workflow of the process is depicted in Figure 4.13.

4.2.5 DAEDALUS

The DAEDALUS (DNA Origami Sequence Design Algorithm for User-defined Structures) [98] project is maintained by the Laboratory for Computational Biology and Biophysics at MIT[11] by Mark Bathe and colleagues. The ul-

[11] http://lcbb.mit.edu/

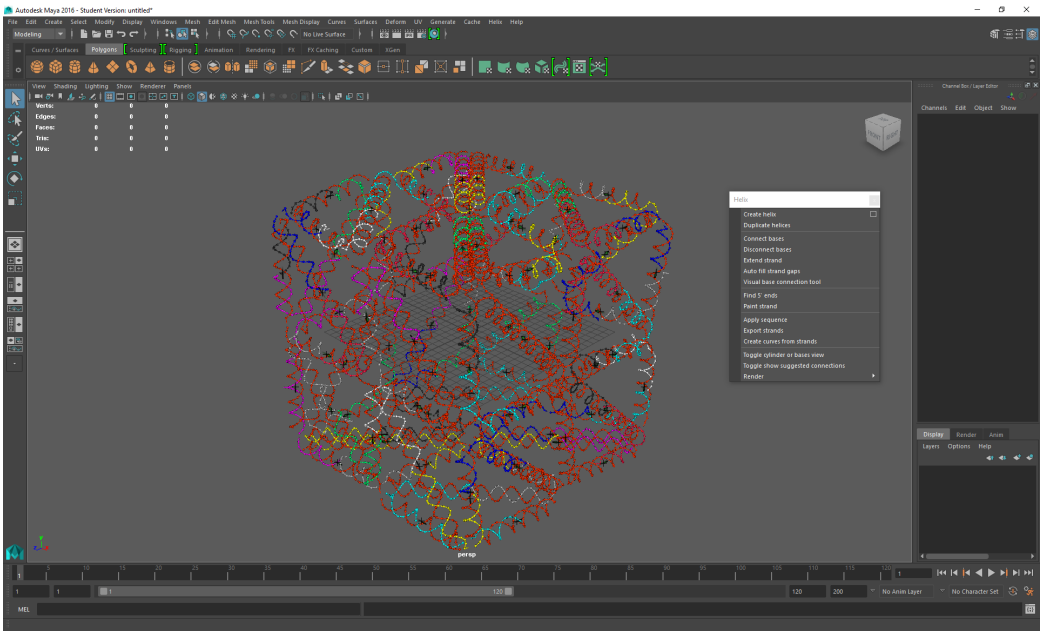

Figure 4.12: 3D atomic model of the designed structure.

timate goal of DAEDALUS is to allow anyone (including non-experts of DNA nanotechnology) to design and synthesize their own DNA assemblies. DAEDALUS is fully automated and supports almost any target 3D geometry. Only an input CAD file is required, from which DAEDALUS will generate the necessary DNA sequences that are required to create scaffolded DNA origami (as explained earlier).

As vHelix, DAEDALUS is also based on the wireframe approach, where the DNA sequences follow the edges of the 3D object. The differences between vHelix and DAEDALUS are that in the latter human input is reduced to a minimum and it also supports greater amount of different sizes and topologies [64]. In addition, DAEDALUS is based on the more rigid double-crossover (DX) edges, whereas vHelix only supports single duplex edges. Furthermore, DAEDALUS does not require expensive software in order to operate (compare to *e.g.* vHelix that requires Maya to operate).

The sequence design procedure for the scaffolded DNA origami is demonstrated in Figure 4.14. By using the given target mesh, the graph of the target structure is constructed (Fig. 4.14(i)), which includes vertex, edge and face information. Next, a spanning tree algorithm is used to generate a route for the scaffold strand to run through the target shape while ensuring that the scaffold does not intersect itself at vertices (Fig. 4.14(ii-iii)). After

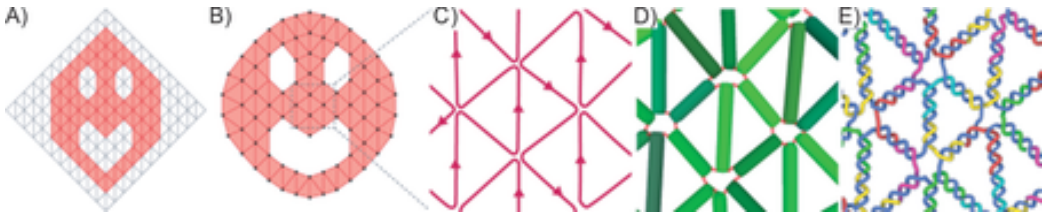

Figure 4.13: (a-b) In order to create a 2D DNA structure, an arbitrary mesh is created. (c) A scaffold path is generated programmatically using a 3D modeling software. (d) A physical simulation is used to generate a DNA model. (e) Finally, the DNA model is imported to vHelix for the staple strand and sequence generation. [9]

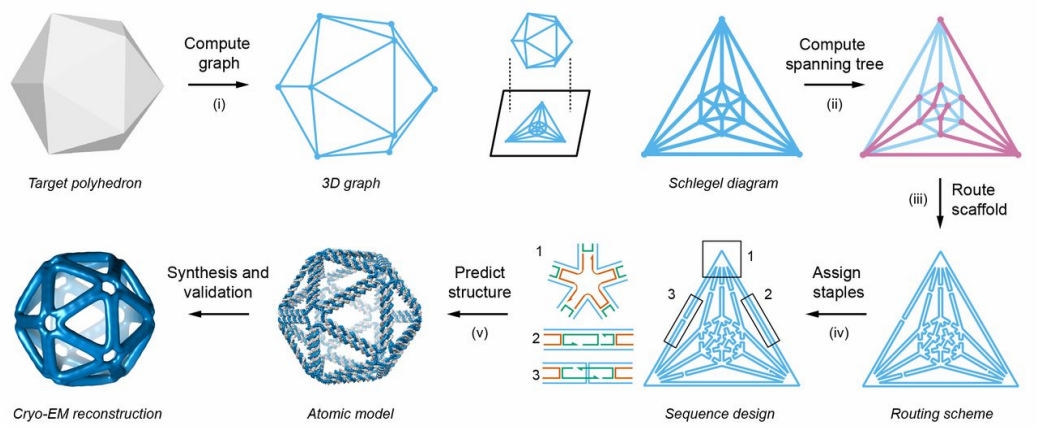

Figure 4.14: DAEDALUS sequence design workflow [98].

the scaffold routing is determined, staple strands are generated automatically (Fig. 4.14(iv)), and finally a 3D atomic model is generated to predict the final structure (Fig. 4.14(v)).

There exist two ways to use DAEDALUS: using the online version[12] or running the software locally on MATLAB. The MATLAB version is demonstrated in this thesis, and as an example, the same cube that was demonstrated with vHelix, is used also here as an input model. The software package can be downloaded from DAEDALUS online portal and it contains good documentation as well. The package is divided into three folders: 'Automated_Design' contains the actual code that operates DAEDALUS, 'Example_Icosahedron' contains pre-generated examples and 'PLY_Files' is the

[12]http://daedalus-dna-origami.org

folder containing several example models that may be used. After setting up the environment, the exemplary script (demo_daedalus.m) provided by the package contains all the necessary code to generate DNA origamis.

First, in order for the script to work properly, current MATLAB path should be set to DAEDALUS root folder ('DAEDALUS_Software'). Next, the name of a text version (not binary) of an input PLY model should be set to the variable 'fname_no_ply' as demonstrated in the script. At this point the minimum amount of base pairs per edge can be adjusted (default is 31 bp and it should be a multiple of 10.5). Once the function 'ply_to_input' has been run and the design variables created, it is possible to modify them manually. For example, individual edge lengths could be changed by modifying the variables in the MATLAB workspace.

Next, the automated design algorithm generates a list of staple strands based on the input variables set in the previous step. The positions and orientations of each nucleotide pairs is also generated. By default, the usual M13mp18 scaffold sequence is used, but it could be changed as well at this stage. This step also generates five different files (including comma separated list of staple sequences) along with three different figures as shown in Figure 4.15. The files generated include spatial and routing information, some debugging variables, a text file visualizing edge sequences and crossovers, CSV file containing staple sequences and CanDo (.cndo) formatted file containing information for CanDo simulation (although CanDo does not support this format at the time of writing this thesis).

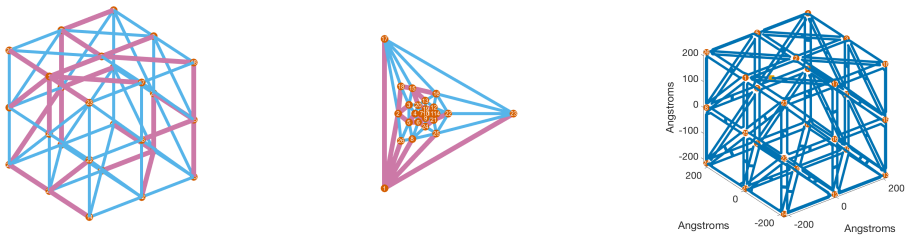

Figure 4.15: On the left the input cube graph (blue lines) and corresponding spanning tree (pink lines). In the middle a Schlegel diagram and on the right the final routing of the scaffold.

Finally, an atomic model of the design can be generated. By default this is disabled, but it can be enabled by setting 'yes_PDB' value to 1. At this point it is also possible to choose to render the atomic model as TIFF image.

Image rendering requires that the visualization software UCSF Chimera[13] is installed and that the path to the install location is specified in the script. Once the atomic model is generated, related files are placed under a folder named after the title of the original model. This folder contains a BILD (positions and orientations of the nucleotides) and a PDB file (atomic model of the structure) that can be opened with UCSF Chimera. Additionally, if image rendering is enabled, this folder also contains three TIFF images of different view angles. The atomic model of the designed cube is shown in Figure 4.16.

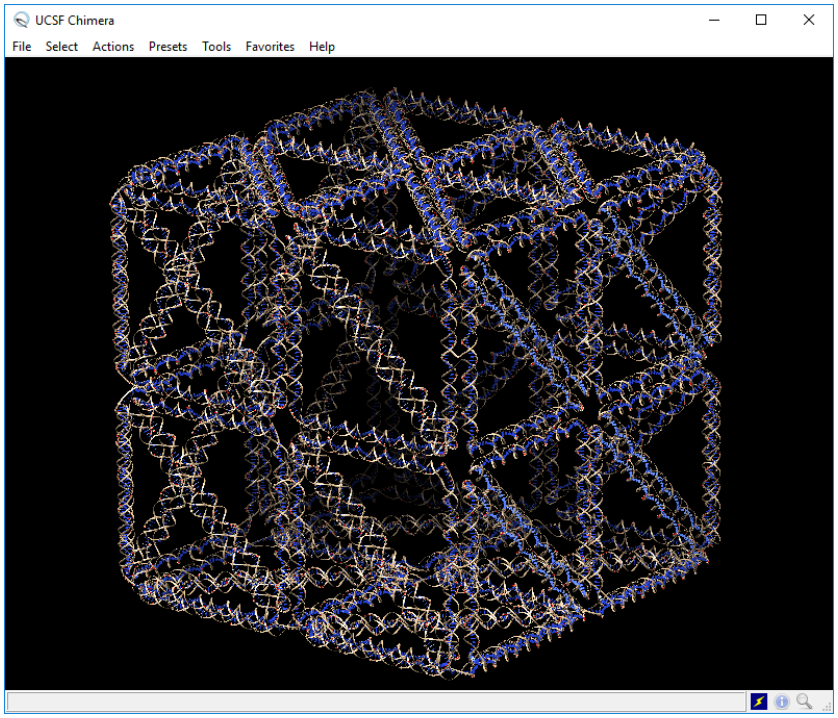

Figure 4.16: Atomic model of the designed cube in Chimera. All the edges are formed by DX molecules.

It is worth mentioning that the generation of an atomic model is a time-demanding process, which might take less than a minute or several hours depending on the size and complexity of the object. For example, it took three minutes to generate the cube, but once the size and amount of edges were doubled, the atomic model generation time increased to 1.5 hours. Large and complex models are heavy to open with UCSF Chimera and thus they require a powerful computer.

[13]https://www.cgl.ucsf.edu/chimera/

4.3 Results

The main results of the comparison presented in this chapter are summarized, commented and listed in the Table 4.1. By looking at the table, it should be easy for a reader to decide which software to use for a given task. A summary of the input and export formats of the used software are summarized and briefly described in the Table 4.2.

A note about wireframe-based software: vHelix supports only single duplex edges and spherical topologies, whereas DAEDALUS enables more rigid double-crossover edges and a structure design with non-spherical topologies. However, the authors of the original vHelix article have recently expanded their design to 2D designs based on flat-sheet meshes [9].

All the software accept customized scaffold lengths and sequences, but only the original DAEDALUS article provides methods to produce (linear) custom-length scaffold strands.

Table 4.1: Compared software features

Functionality	**caDNAno**	**Tiamat**	**vHelix**	**DAEDALUS**
Platform				
Windows	YES	YES	YES	YES
OS X	YES[14]	NO	YES	YES
Linux	NO	NO	YES	YES
Online	NO	NO	NO	YES
Top-down	NO	NO	YES	YES
Shape space				
2D	YES	YES	YES	YES
3D	YES	YES	YES	YES
Lattice-free	NO	YES	YES	YES
Wireframe	NO	YES	YES[15]	YES[16]
Twists/bends	YES[17]	NO	NO	NO
Visualization	YES[18]	YES[19]	YES[20]	YES[21]
Automation				
Scaffold routing	NO	NO[22]	YES[23]	YES[24]
Staple design	YES[25]	NO[22]	YES[26]	YES[27]

[14]No support for OS X El Capitan.

[15]Wireframe with single duplex edges.

[16]Wireframe with DX molecule edges.

[17]Structures can be verified by CanDo simulation.

[18]Old versions have the option of direct visualization, but the visualization in the new versions is only available if Maya is installed. Structures can be verified by CanDo simulation after design.

[19]Direct visualization and possibility for CanDo simulation.

[20]Direct visualization with Maya.

[21]Direct visualization combined with the atomic model. Moreover, the actual solution shapes can be further verified by CanDo simulation.

[22]Structures are created by combining dsDNA motifs together. However, the software generates a list of staple sequences as an output.

[23]Mesh size can be tuned by scaling factor before scaffold routing.

[24]Mesh size (edge length) can be modified in MATLAB code.

[25]Creates staples automatically, but the strands have to be cut manually or by using the autobreak option (works to some extent). However, staples can be modified manually in an intuitive way.

[26]Staples can be also modified by hand and they are visualized in Maya.

[27]It is challenging to modify individual staples by hand once the structure is created.

Table 4.2: Export formats

Software	Input	Output
caDNAno	.json	.json, .csv
	caDNAno-generated files (.json) can be imported and modified.	Strand sequences are exported as .csv. Simulation can be carried out by CanDo using .json files as an input.
Tiamat	.dna	.dna, .avi, .txt
	Tiamat-generated files (.dna) can be imported and modified.	Strand sequences are exported as .txt. Cartoons can be viewed as .avi files. Simulation can be carried out by CanDo using .dna files as an input.
vHelix	.rpoly	.csv
	Routing information generated by BSCOR software is imported to vHelix as .rpoly.	Strand sequences are exported as .csv. Simulation of the designs is currently under development.
DAEDALUS	.ply, .mat	.csv, .cndo, .tif, .pdb, .bild
	Models are read from .ply files. Previously generated routing information can be imported from .mat file.	Staple sequences are exported as .csv, and .cndo files can be used in CanDo simulation. Atomic models can be exported as .tif images or .pdb and .bild files that can be read by Chimera.
CanDo	.json, .dna, .cndo, .csv	.pdb, .bild
	Generated .csv files (sequences) can be uploaded along with the design files in order to produce the atomic models.	Cylindrical and atomic models can be exported as .pdb and .bild files that can be read by Chimera.

Chapter 5

Implementation of a plugin for caDNAno

In this chapter a plugin for speeding up a design of structures having different helix rise is presented. Here, Kostiainen's research group needed to design a RNA/DNA rectangle (without a twist) using caDNAno software. In more detail, the scaffold strand in this structure is a long linear RNA molecule that they want to fold into a desired shape with the help of short DNA staples (see the next chapter). However, the usual helical rise in RNA/DNA duplex is 11 bp/turn and therefore a manual base pair extension in the caDNAno design is required.

In general, caDNAno design lattice is divided to either 8 bases segments (square lattice) or 7 bases (honeycomb). This is due to the fact that in square lattice geometry 8 bases between crossovers equals 274.3 degrees, which is close to the ideal angle (270 degrees). However, due to the small mismatch in angles, a slight right-handed twist occurs without base relaxation. In a honeycomb lattice, for one, dividing the lattice by the segments of 7 bases results in 240 degrees angle between neighboring crossovers, which exactly matches the ideal angle (240 degrees). These calculations assume that dsDNA segments adopt a 10.5 bp/turn rule.

Now, if one needs to design an abovementioned RNA/DNA hybrid origami structure with 11 bp/turn using caDNAno software, one needs to add an extra base pair for example at the intervals of 32 bases. This results in 33 base pairs per 3 full helical turns (on average 11 bp/turn) and thus, this extra base pair should be able to cancel out the twist of the structure. The next section describes the method, the code and the instructions in detail.

5.1 Development process

Running caDNAno from the source code is a quite tedious process, because the required dependencies are outdated and not straightforward to install. The process starts by downloading caDNAno source files from Github and installing Qt 4.7 and PyQt. Qt is a cross-platform application development framework for developing graphical user interfaces, and PyQt offers Python bindings for the Qt framework. Furthermore, it should be verified that the installed Python version should not be newer than 2.7, since otherwise the application would not launch properly.

Once all the dependencies are installed and working, the main interface is launched by issuing command 'python main.py' from the terminal. Plugins directory contains the plugins that will be loaded at the application startup. That directory also contains the autobreak plugin as an example. The easiest way to start developing a plugin is to make a copy of the autobreak plugin and begin to modify that. The plugin is loaded from the '__init__.py' file and added to the menu from there. caDNAno code is not thoroughly documented and therefore finding the fundamental parts of the application code is a time-consuming task. The final code that generates extra base pairs is presented below:

```
#1 import cadnano, util
#2 util.qtWrapImport('QtGui', globals(), ['QIcon', 'QPixmap', \
#3    'QAction'])
#4
#5 class ExtraBpHandler(object):
#6    def __init__(self, document, window):
#7        self.doc, self.win = document, window
#8        icon10 = QIcon()
#9        icon10.addPixmap(QPixmap(":/pathtools/insert"), \
#10           QIcon.Normal, QIcon.Off)
#11       self.actionExtraBp = QAction(window)
#12       self.actionExtraBp.setIcon(icon10)
#13       self.actionExtraBp.setText('ExtraBp')
#14       self.actionExtraBp.setToolTip("Click this button to \
#15           insert extra base pairs.")
#16       self.actionExtraBp.setObjectName("actionExtraBp")
#17       self.actionExtraBp.triggered.connect(self \
#18           .actionExtraBpSlot)
#19       self.win.menuPlugins.addAction(self.actionExtraBp)
#20       # add to main tool bar
```

```
#21         self.win.topToolBar.insertAction(self.win \
#22            .actionFiltersLabel, self.actionExtraBp)
#23         self.win.topToolBar.insertSeparator(self.win \
#24            .actionFiltersLabel)
#25         self.configDialog = None
#26
#27     def actionExtraBpSlot(self):
#28         part = self.doc.controller().activePart()
#29         if part != None:
#30             for vh in part.getVirtualHelices():
#31                 scafSS = vh.scaffoldStrandSet()
#32                 for strand in scafSS:
#33                     lo, hi = strand.idxs()
#34                     numOfBp = ((hi - lo) // 32) + 1
#35                     for x in range(0, numOfBp):
#36                         strand.addInsertion(lo + (x * 32), 1)
#37
#38 def documentWindowWasCreatedSlot(doc, win):
#39     doc.extraBpHandler = ExtraBpHandler(doc, win)
#40
#41 # Initialization
#42 for c in cadnano.app().documentControllers:
#43     doc, win = c.document(), c.window()
#44     doc.extraBpHandler = ExtraBpHandler(doc, win)
#45 cadnano.app().documentWindowWasCreatedSignal. \
#46     connect(documentWindowWasCreatedSlot)
```

Lines 1-25 and 38-46 are used for initialization and adding the plugin into the caDNAno user interface. All the important code is located on lines 27-36, where it can be seen that each individual strand is looped through, and an extra base pair is added at the intervals of 32 bases. On the line 30 each helix is read into variable 'vh', and after that, on the line 32 each strand is read from that helix into variable 'strand'. Line 33 finds the lowest and highest positions in the current strand and the line 34 counts the amount of base pairs per that strand. Finally, on line 36 a base pair is added every 32 bases.

Alternative code that is presented below follows the same principle, but inserts the extra base pairs exactly in the same column positions (stored in variable 'initPos' on the line 4) at each row, and the 'offset' where the first base pair is added can be chosen on the line 5. This code is used to create extra base pairs in hybrid 64HB structure (Fig. 6.3). However, this latter

version might not work properly if there are crossovers in the middle of the design (overcompensates the twist). In that situation, the first code might function better. The plugin should be further developed in order to allow the user to choose how to insert base pairs and what offset should be used. However, the offsets and rules can be easily changed in the code by the user.

```
#1    def actionExtraBpSlot(self):
#2        part = self.doc.controller().activePart()
#3        if part != None:
#4            initPos = -1
#5            offset = 15
#6            for vh in part.getVirtualHelices():
#7                scafSS = vh.scaffoldStrandSet()
#8                for strand in scafSS:
#9                    lo, hi = strand.idxs()
#10                    if initPos == -1:
#11                        initPos = lo + offset
#12                    numOfBp = ((hi - lo) // 32) + 1
#13                    for x in range(0, numOfBp):
#14                        strand.addInsertion(initPos \
#15                            + (x * 32), 1)
```

5.2 Installation and usage instructions

Installation of a plugin is rather straightforward: the provided folder containing the plugin should be placed under the plugins folder in the caDNAno installation directory. The plugin adds itself to the top bar next to the 'Autobreak' button, but it can also be run from the Plugins menu. Once the design is created in caDNAno, pressing the ExtraBp button inserts extra base pairs automatically to the design. Functionality is the same as if extra bases would have been inserted manually one by one, but the plugin removes the manual work. This feature could be extremely useful for larger designs, where the linear scaffold routing goes simply back and forth.

Chapter 6

Evaluation of the plugin

6.1 Design of a hybrid RNA-DNA rectangle

The structure was designed using caDNAno in a square lattice geometry. However, the scaffold strand used here is linear, and it is made of RNA instead of DNA. This leads to a slightly different helical rise, which has to be taken into account as explained above. Since the scaffold is linear, it was designed simply to go back-and-forth in the structure by going through 28 helices (with 34-nucleotide (nt) RNA loops at the ends where the scaffold turns around). Now, by using the aforementioned plugin, it results in an extra base pair after each 32 bp in the design as seen in Figure 6.1. The staples are designed in such a way that they usually contain only one crossover position (and in addition, the crossovers are made of only only staple (half of the full Holliday junction is always missing)). This kind of design expectedly makes the structure weaker, but on the other hand, this should enable easier relaxation of the possible strain.

Moreover, all the uracil bases (U) in the RNA-scaffold sequence had to be replaced with thymines (T) in order to get correct complementary DNA strands as an output. This custom scaffold sequence was inserted using a sequence-tool of caDNANo, and subsequently, the software generated the staple sequences.

6.2 CanDo verification

RNA-DNA hybrid helix has approximately 11 bp per turn, and its mechanical properties may differ from those of double-stranded DNA. Here, in the CanDo simulation all the default parameters were kept the same, except the geometry parameter 'Crossover spacing [bp]' (see Fig. 4.5), which was

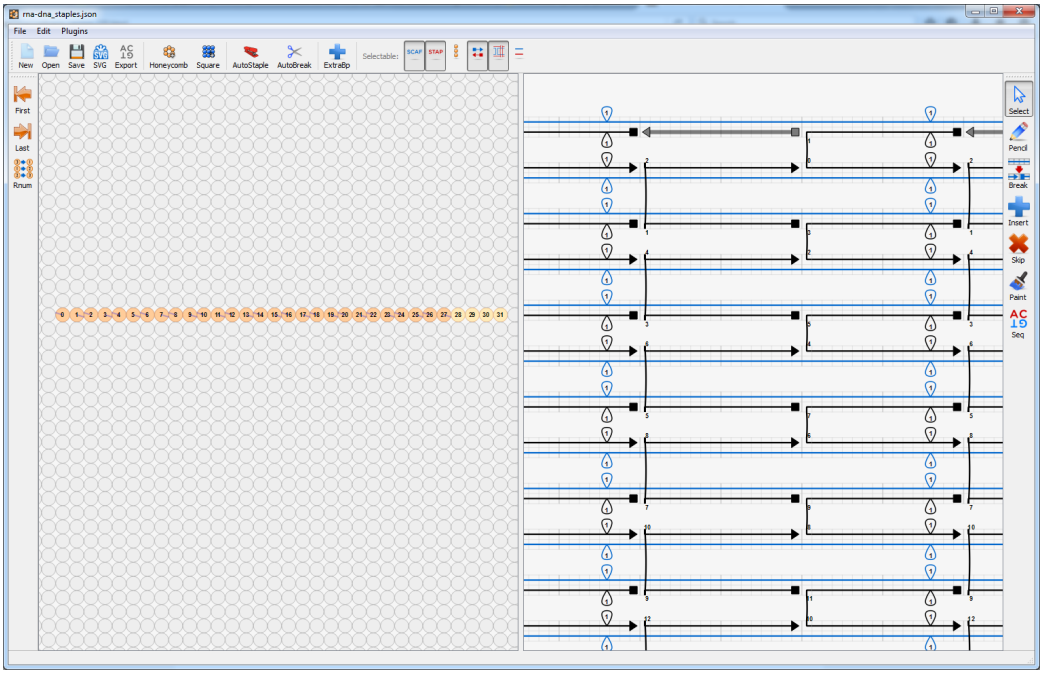

Figure 6.1: Extra base pairs (after each 32 bp) are added to the rectangular origami in caDNAno.

changed from the default 10.5 to 11 in order to achieve a more realistic simulation result. The simulated shape in Figure 6.2 shows that the designed rectangle forms correctly and without undesired twisting.

6.3 Real life verification

By using the the abovementioned design (caDNAno and the plugin) and a linear 7339 nt long RNA sequence as an input, the set of 188 unique DNA staple strand sequences was generated. Finally, this set of strands was purchased from Integrated DNA Technologies (IDT). Unfortunately, the RNA scaffold synthesis took longer than expected, and thus the actual structure has not yet been folded. Therefore, the real life verification of the design was not available when writing this thesis. Nevertheless, if the structure could fold properly in the laboratory, it would be a further proof that the design, the code and the other assumptions were appropriate.

Figure 6.2: Two views of the CanDo simulated solution shape of the hybrid RNA-DNA rectangle (with extra base pairs). The structure has been simulated assuming 11 bp per full helical turn.

6.4 3D construction

To test if the same design rules and the plugin could be generalized to 3D fabrication, a hybrid RNA-DNA brick was created by using caDNAno (Fig. 6.3). The design is a 64-helix bundle (64HB) in square lattice, but it is not exactly the same as the one presented in Chapter 4 (Fig. 4.7), since here the scaffold is linear and the routing is carried out in a back-and-forth manner (no scaffold crossovers).

The plugin presented above (extra base pairs in the same column) was used to add the extra base pairs to the design, and again, the structure was verified by simulating it with CanDo. It can be seen that the structure appears relatively straight (Fig. 6.4). This result indicates that the addition of

the extra base pairs averages out the undesired twist. It is important to notice here that since the staple design was created using autobreak option, the number of full Holliday junctions formed by the staples (double-crossovers) is maximized opposed to the RNA-DNA rectangle. This presumably makes the structure more rigid and it restricts the relaxation of the twists/bends.

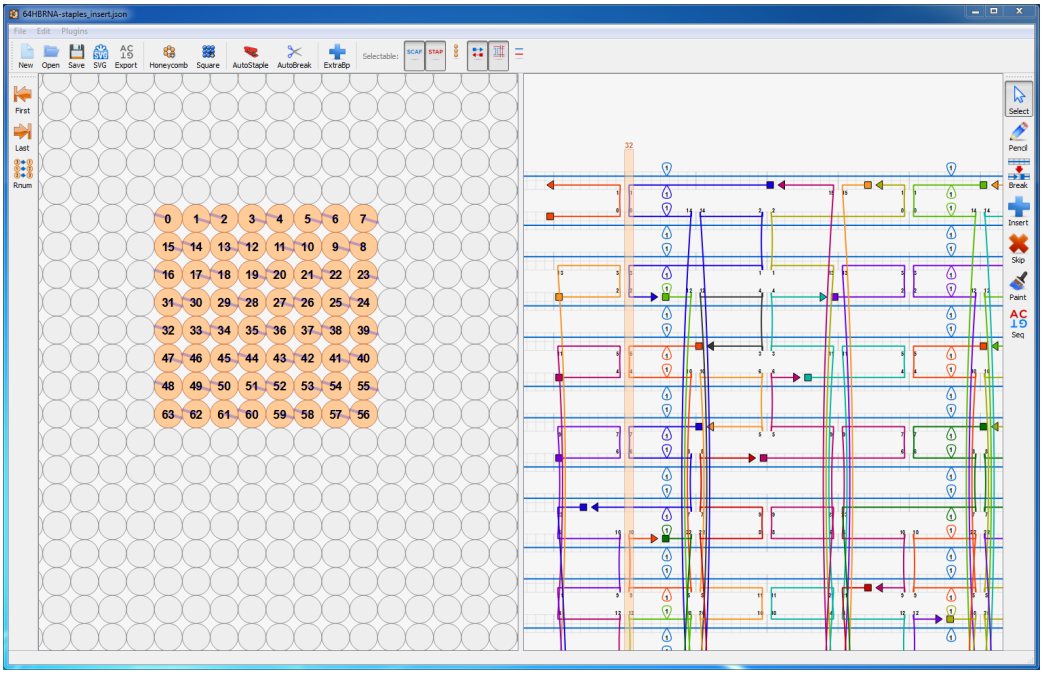

Figure 6.3: Extra base pairs (after each 32 bp) are added to the 64HB structure in caDNAno.

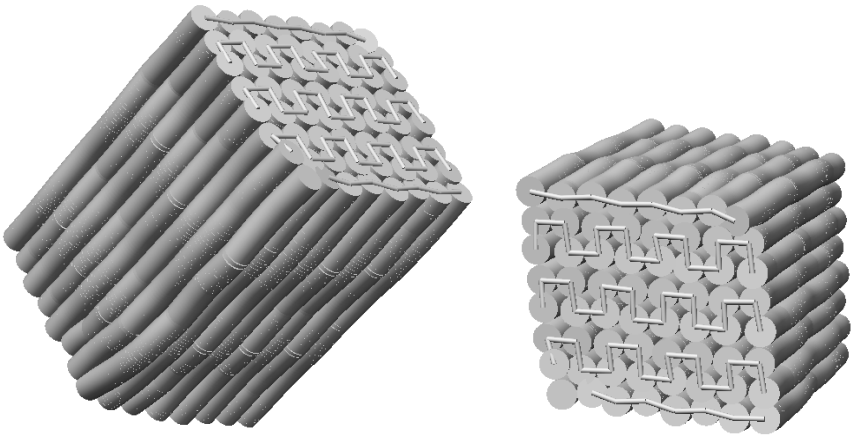

Figure 6.4: Two views of the CanDo simulated solution shape of the hybrid RNA-DNA 64HB (with extra base pairs). The structure has been simulated assuming 11 bp per full helical turn.

Chapter 7

Conclusions and future perspectives

The techniques for assembling DNA molecules into desired 3D structures have advanced greatly in the past decade, and currently the field is attracting a lot of people from different backgrounds. In order to design even more complex structures, powerful design software are needed. A recent approach where the design process begins by introducing a desired 3D shape first has simplified and speeded up the task. This is a so-called top-down method (as opposed to bottom-up approach) and in this procedure the scaffold routing and staple design is fully automated (at least with the tested software in this thesis) requiring as little human input as possible. Furthermore, this novel top-down method allows nanostructures of almost any shapes and sizes.

In this thesis the fundamentals of DNA nanotechnology and DNA origami technique were discussed. Furthermore, a brief history to the field was given and some of the prominent applications were shortly mentioned as well. The main focus was on the comparison of design software. The tested programs for comparison were chosen mainly on the basis of novelty and the estimated amount of users. Moreover, the compared software were the only reasonable ones that were available at the time of writing this thesis. Example structures (bricks and cubes) were designed with the software, and based on the findings during the process, the results and features of the software were gathered in a single table. The idea of the table is to function as a guide for which software to use for particular design needs. This thesis should also work as a tutorial for using the introduced programs. All the steps required for designing the example structures are documented in this thesis. Additionally, a plugin for caDNAno was developed and evaluated. The idea of the plugin was to simplify the design of RNA-DNA hybrids in caDNAno by inserting extra base pairs automatically in the correct positions. On the other hand,

it serves as a starting point for any users to add their own plugins to the software.

One of the future goals in software development would be to increase the durability of the structures, for example, by allowing more sophisticated edge design. Currently, closed-surface topologies are difficult to construct using the top-down methods. Another problem is that the generated nanostructures may be troublesome to modify afterwards. For example, vHelix allows only staple strands of the generated structure to be modified, and with a current version of DAEDALUS any modification is virtually impossible. In addition, as the rationally designed DNA objects are getting more and more involved in the living systems, to realize an automated method for designing and creating structures directly *in vivo* would be a huge step forward for many bio-oriented applications [23, 64].

Bibliography

[1] Editorial: The unnatural order of things. *Nature Nanotechnology 4*, 4 (2009), 203.

[2] Amir, Y., Ben-Ishay, E., Levner, D., Ittah, S., Abu-Horowitz, A., and Bachelet, I. Universal computing by DNA origami robots in a living animal. *Nature Nanotechnology 9*, 5 (2014), 353–357.

[3] Andersen, E. S., Dong, M., Nielsen, M. M., Jahn, K., Lind-Thomsen, A., Mamdouh, W., Gothelf, K. V., Besenbacher, F., and Kjems, J. DNA origami design of dolphin-shaped structures with flexible tails. *ACS Nano 2*, 6 (2008), 1213–1218. Software from: `http://cdna.au.dk/software/`. Accessed 7 Apr 2016.

[4] Andersen, E. S., Dong, M., Nielsen, M. M., Jahn, K., Subramani, R., Mamdouh, W., Golas, M. M., Sander, B., Stark, H., Oliveira, C. L. P., Pedersen, J. S., Birkedal, V., Besenbacher, F., Gothelf, K. V., and Kjems, J. Self-assembly of a nanoscale DNA box with a controllable lid. *Nature 459*, 7243 (2009), 73–76.

[5] Bai, X.-c., Martin, T. G., Scheres, S. H. W., and Dietz, H. Cryo-EM structure of a 3D DNA-origami object. *Proceedings of the National Academy of Sciences 109*, 49 (2012), 20012–20017.

[6] Bansal, M. DNA structure: Revisiting the Watson-Crick double helix. *Current Science 85*, 11 (2003), 1556–1563.

[7] Barish, R. D., Schulman, R., Rothemund, P. W. K., and Winfree, E. An information-bearing seed for nucleating algorithmic self-assembly. *Proceedings of the National Academy of Sciences 106*, 15 (2009), 6054–6059.

[8] Bell, N. A., Engst, C. R., Ablay, M., Divitini, G., Ducati, C., Liedl, T., and Keyser, U. F. DNA origami nanopores. *Nano Letters 12*, 1 (2012), 512–517.

[9] Benson, E., Mohammed, A., Bosco, A., Teixeira, A. I., Orponen, P., and Höbgerg, B. Computer-aided production of scaffolded DNA nanostructures from flat sheet meshes. *Angewandte Chemie International Edition 55*, 31 (2016), 8869–8872.

[10] Benson, E., Mohammed, A., Gardell, J., Masich, S., Czeizler, E., Orponen, P., and Högberg, B. DNA rendering of polyhedral meshes at the nanoscale. *Nature 523*, 7561 (2015), 441–444.

[11] Birac, J. J., Sherman, W. B., Kopatsch, J., Constantinou, P. E., and Seeman, N. C. Architecture with GIDEON, a program for design in structural DNA nanotechnology. *Journal of Molecular Graphics and Modelling 25*, 4 (2006), 470–480.

[12] Brenneman, A., and Condon, A. Strand design for biomolecular computation. *Theoretical Computer Science 287*, 1 (2002), 39–58. Natural Computing.

[13] Castro, C. E., Kilchherr, F., Kim, D.-N., Shiao, E. L., Wauer, T., Wortmann, P., Bathe, M., and Dietz, H. A primer to scaffolded DNA origami. *Nature Methods 8*, 3 (2011), 221–229.

[14] Church, G. M., Gao, Y., and Kosuri, S. Next generation digital information storage in DNA. *Science 337*, 6102 (2012), 1628.

[15] DeLano, W. L. The PyMOL molecular graphics system, 2002. Software URL: `http://www.pymol.org/`. Accessed 7 Apr 2016.

[16] Derr, N. D., Goodman, B. S., Jungmann, R., Leschziner, A. E., Shih, W. M., and Reck-Peterson, S. L. Tug-of-war in motor protein ensembles revealed with a programmable DNA origami scaffold. *Science 338*, 6107 (2012), 662–665.

[17] Dietz, H., Douglas, S. M., and Shih, W. M. Folding DNA into twisted and curved nanoscale shapes. *Science 325*, 5941 (2009), 725–730.

[18] Dirks, R. M., Lin, M., Winfree, E., and Pierce, N. A. Paradigms for computational nucleic acid design. *Nucleic Acids Research 32*, 4 (2004), 1392–1403.

[19] Douglas, S. M., Bachelet, I., and Church, G. M. A logic-gated nanorobot for targeted transport of molecular payloads. *Science 335*, 6070 (2012), 831–834.

[20] Douglas, S. M., Chou, J. J., and Shih, W. M. DNA-nanotube-induced alignment of membrane proteins for NMR structure determination. *Proceedings of the National Academy of Sciences 104*, 16 (2007), 6644–6648.

[21] Douglas, S. M., Dietz, H., Liedl, T., Hogberg, B., Graf, F., and Shih, W. M. Self-assembly of DNA into nanoscale three-dimensional shapes. *Nature 459*, 7245 (2009), 414–418.

[22] Douglas, S. M., Marblestone, A. H., Teerapittayanon, S., Vazquez, A., Church, G. M., and Shih, W. M. Rapid prototyping of 3D DNA-origami shapes with caDNAno. *Nucleic Acids Research 37*, 15 (2009), 5001–5006. Software available at: `http://cadnano.org/`. Accessed 14 Apr 2016.

[23] Elbaz, J., Yin, P., and Voigt, C. A. Genetic encoding of DNA nanostructures and their self-assembly in living bacteria. *Nature Communications 7* (2016), 11179.

[24] Endo, M., Yamamoto, S., Tatsumi, K., Emura, T., Hidaka, K., and Sugiyama, H. RNA-templated DNA origami structures. *Chemical Communications 49*, 28 (2013), 2879–2881.

[25] Eskelinen, A.-P., Moerland, R. J., Kostiainen, M. A., and Törmä, P. Self-assembled silver nanoparticles in a bowtie antenna configuration. *Small 10*, 6 (2014), 1057–1062.

[26] Eskelinen, A.-P., Rosilo, H., Kuzyk, A., Törmä, P., and Kostiainen, M. A. Controlling the formation of DNA origami structures with external signals. *Small 8*, 13 (2012), 2016–2020.

[27] Feldkamp, U., and Niemeyer, C. M. Rational Design of DNA Nanoarchitectures. *Angewandte Chemie International Edition 45*, 12 (2006), 1856–1876.

[28] Feynman, R. P. There's plenty of room at the bottom. *Engineering and Science 23*, 5 (1960), 22–36.

[29] Fu, T. J., and Seeman, N. C. DNA double-crossover molecules. *Biochemistry 32*, 13 (1993), 3211–3220.

[30] Geary, C., Rothemund, P. W. K., and Andersen, E. S. A single-stranded architecture for cotranscriptional folding of RNA nanostructures. *Science 345*, 6198 (2014), 799–804.

[31] Gerling, T., Wagenbauer, K. F., Neuner, A. M., and Dietz, H. Dynamic DNA devices and assemblies formed by shape-complementary, non–base pairing 3D components. *Science 347*, 6229 (2015), 1446–1452.

[32] Goldman, N., Bertone, P., Chen, S., Dessimoz, C., LeProust, E. M., Sipos, B., and Birney, E. Towards practical, high-capacity, low-maintenance information storage in synthesized DNA. *Nature 494*, 7435 (2013), 77–80.

[33] Gopinath, A., Miyazono, E., Faraon, A., and Rothemund, P. W. K. Engineering and mapping nanocavity emission via precision placement of DNA origami. *Nature 535*, 7612 (2016), 401–405.

[34] Gopinath, A., and Rothemund, P. W. K. Optimized assembly and covalent coupling of single-molecule DNA origami nanoarrays. *ACS Nano 8*, 12 (2014), 12030–12040.

[35] Grabow, W. W., and Jaeger, L. RNA self-assembly and RNA nanotechnology. *Accounts of Chemical Research 47*, 6 (2014), 1871–1880.

[36] Guo, P. The emerging field of RNA nanotechnology. *Nature Nanotechnology 5*, 12 (2010), 833–842.

[37] Han, D., Pal, S., Nangreave, J., Deng, Z., Liu, Y., and Yan, H. DNA origami with complex curvatures in three-dimensional space. *Science 332*, 6027 (2011), 342–346.

[38] Han, D., Pal, S., Yang, Y., Jiang, S., Nangreave, J., Liu, Y., and Yan, H. DNA gridiron nanostructures based on four-arm junctions. *Science 339*, 6126 (2013), 1412–1415.

[39] Ho, P. S., and Carter, M. DNA structure: Alphabet soup for the cellular soul. In *DNA Replication-Current Advances*, H. Seligmann, Ed. 2011, ch. 1.

[40] Hong, F., Jiang, S., Wang, T., Liu, Y., and Yan, H. 3D framework DNA origami with layered crossovers. *Angewandte Chemie Internation Edition 55*, 41 (2016), 12832–12835.

[41] Jones, M. R., Seeman, N. C., and Mirkin, C. A. Programmable materials and the nature of the DNA bond. *Science 347*, 6224 (2015), 1260901.

[42] Jungmann, R., Avendaño, M. S., Woehrstein, J. B., Dai, M., Shih, W. M., and Yin, P. Multiplexed 3D cellular super-resolution imaging with DNA-PAINT and exchange-PAINT. *Nature Methods 11*, 3 (2014), 313–318.

[43] Ke, Y., Douglas, S. M., Liu, M., Sharma, J., Cheng, A., Leung, A., Liu, Y., Shih, W. M., and Yan, H. Multilayer DNA origami packed on a square lattice. *Journal of the American Chemical Society 131*, 43 (2009), 15903–15908.

[44] Ke, Y., Ong, L. L., Shih, W. M., and Yin, P. Three-dimensional structures self-assembled from DNA bricks. *Science 338*, 6111 (2012), 1177–1183.

[45] Kershner, R. J., Bozano, L. D., Micheel, C. M., Hung, A. M., Fornof, A. R., Cha, J. N., Rettner, C. T., Bersani, M., Frommer, J., Rothemund, P. W. K., and M, W. G. Placement and orientation of individual DNA shapes on lithographically patterned surfaces. *Nature Nanotechnology 4*, 9 (2009), 557–561.

[46] Kim, D.-N., Kilchherr, F., Dietz, H., and Bathe, M. Quantitative prediction of 3D solution shape and flexibility of nucleic acid nanostructures. *Nucleic Acids Research 40*, 7 (2012), 2862–2868.

[47] Kiviaho, J. K., Linko, V., Ora, A., Tiainen, T., Järvihaavisto, E., Mikkilä, J., Tenhu, H., Nonappa, and Kostiainen, M. A. Cationic polymers for DNA origami coating – examining their binding efficiency and tuning the enzymatic reaction rates. *Nanoscale 8*, 22 (2016), 11674–11680.

[48] Korpelainen, V., Linko, V., Seppä, J., Lassila, A., and Kostiainen, M. A. DNA origami structures as calibration standards for nanometrology. *Measurement Science and Technology* (2016), in print.

[49] Krishnan, S., Ziegler, D., Arnaut, V., Martin, T. G., Kapsner, K., Henneberg, K., Bausch, A. R., Dietz, H., and Simmel, F. C. Molecular transport through large-diameter DNA nanopores. *Nature Communications 7*, 12787 (2016).

[50] Kuzyk, A., Schreiber, R., Fan, Z., Pardatscher, G., Roller, E.-M., Högele, A., Simmel, F. C., Govorov, A. O., and Liedl, T. DNA-based self-assembly of chiral plasmonic nanostructures with tailored optical response. *Nature 483*, 7389 (2012), 311–314.

[51] Kuzyk, A., Schreiber, R., Zhang, H., Govorov, A. O., Liedl, T., and Liu, N. Reconfigurable 3D plasmonic metamolecules. *Nature Materials 13*, 9 (2014), 862–866.

[52] Kuzyk, A., Yurke, B., Toppari, J. J., Linko, V., and Törmä, P. Dielectrophoretic trapping of DNA origami. *Small 4*, 13 (2008), 447–450.

[53] Laboratory for Computational Biology & Biophysics, MIT. CanDo: Computer-aided engineering for DNA origami. Webpage. `http://cando-dna-origami.org/`. Accessed 14 Apr 2016.

[54] Langecker, M., Arnaut, V., Martin, T. G., List, J., Renner, S., Mayer, M., Dietz, H., and Simmel, F. C. Synthetic lipid membrane channels formed by designed DNA nanostructures. *Science 338*, 6109 (2012), 932–936.

[55] Lee, H., Lytton-Jean, A. K. R., Chen, Y., Love, K. T., Park, A. I., Karagiannis, E. D., Sehgal, A., Querbes, W., Zurenko, C. S., Jayaraman, M., Peng, C. G., Charisse, K., Borodovsky, A., Manoharan, M., Donahoe, J. S., Truelove, J., Nahrendorf, M., Langer, R., and Anderson, D. G. Molecularly self-assembled nucleic acid nanoparticles for targeted in vivo siRNA delivery. *Nature Nanotechnology 7*, 6 (2012), 389–393.

[56] Liedl, T., Högberg, B., Tytell, J., Ingber, D. E., and Shih, W. M. Self-assembly of three-dimensional prestressed tensegrity structures from DNA. *Nature Nanotechnology 5*, 7 (2010), 520–524.

[57] Liedl Group. DNA origami. `http://www.softmatter.physik.uni-muenchen.de/liedl_group/index.html`. Accessed 24 Feb 2016.

[58] Lin, C., Jungmann, R., Leifer, A., Li, C., Levner, D., Church, G. M., Shih, W. M., and Yin, P. Submicrometre geometrically encoded fluorescent barcodes self-assembled from DNA. *Nature Chemistry 4*, 10 (2012), 832–839.

[59] Lin, C., Liu, Y., Rinker, S., and Yan, H. DNA Tile Based Self-Assembly: Building Complex Nanoarchitectures. *ChemPhysChem 7*, 8 (2006), 1641–1647.

[60] Lin, C., Perrault, S. D., Kwak, M., Graf, F., and Shih, W. M. Purification of DNA-origami nanostructures by rate-zonal centrifugation. *Nucleic Acids Research* (2012), gks1070.

[61] Linko, V. *DNA-based applications in molecular electronics.* PhD thesis, University of Jyväskylä, 2011. Research report/Department of Physics, 2/2011. `http://urn.fi/URN:ISBN:978-951-39-4259-5`.

[62] Linko, V., and Dietz, H. The enabled state of DNA nanotechnology. *Current Opinion in Biotechnology 24*, 4 (2013), 555–561.

[63] Linko, V., Eerikäinen, M., and Kostiainen, M. A. A modular DNA origami-based enzyme cascade nanoreactor. *Chemical Communications 51*, 25 (2015), 5351–5354.

[64] Linko, V., and Kostiainen, M. A. Automated design of DNA origami. *Nature Biotechnology 34*, 8 (2016), 826–827.

[65] Linko, V., Nummelin, S., Aarnos, L., Tapio, K., Toppari, J. J., and Kostiainen, M. A. DNA-based enzyme reactors and systems. *Nanomaterials 6*, 8 (2016), 139.

[66] Linko, V., Ora, A., and Kostiainen, M. A. DNA nanostructures as smart drug-delivery vehicles and molecular devices. *Trends in Biotechnology 33*, 10 (2015), 586–594.

[67] Linko, V., Shen, B., Tapio, K., Toppari, J. J., Kostiainen, M. A., and Tuukkanen, S. One-step large-scale deposition of salt-free DNA origami nanostructures. *Scientific Reports 5* (2015), 15634.

[68] Linko, V., and Toppari, J. J. Self-assembled DNA-based structures for nanoelectronics. *Journal of Self-Assembly and Molecular Electronics (SAME) 1*, 1 (2013), 101–124.

[69] Martin, T. G., and Dietz, H. Magnesium-free self-assembly of multi-layer DNA objects. *Nature Communications 3* (2012), 1103.

[70] Maune, H. T., Han, S.-P., Barish, R. D., Bockrath, M., Goddard III, W. A., Rothemund, P. W. K., and Winfree, E. Self-assembly of carbon nanotubes into two-dimensional geometries using DNA origami templates. *Nature Nanotechnology 5*, 1 (2010), 61–66.

[71] Mikkilä, J., Eskelinen, A.-P., Niemelä, E. H., Linko, V., Frilander, M. J., Törmä, P., and Kostiainen, M. A. Virus-encapsulated DNA origami nanostructures for cellular delivery. *Nano Letters 14*, 4 (2014), 2196–2200.

[72] Myhrvold, C., Dai, M., Silver, P. A., and Yin, P. Isothermal self-assembly of complex DNA structures under diverse and biocompatible conditions. *Nano Letters 13*, 9 (2013), 4242–4248.

[73] Ora, A., Järvihaavisto, E., Zhang, H., Auvinen, H., Santos, H. A., Kostiainen, M. A., and Linko, V. Cellular delivery of enzyme-loaded DNA origami. *Submitted for publication* (2016).

[74] Pan, K., Kim, D.-N., Zhang, F., Adendorff, M. R., Yan, H., and Bathe, M. Lattice-free prediction of three-dimensional structure of programmed DNA assemblies. *Nature Communications 5* (2014), 5578.

[75] Pinheiro, A. V., Han, D., Shih, W. M., and Yan, H. Challenges and opportunities for structural DNA nanotechnology. *Nature Nanotechnology 6*, 12 (2011), 763–772.

[76] Plesa, C., Ananth, A. N., Linko, V., Gülcher, C., Katan, A. J., Dietz, H., and Dekker, C. Ionic permeability and mechanical properties of DNA origami nanoplates on solid-state nanopores. *ACS Nano 8*, 1 (2014), 35–43.

[77] Qian, L., and Winfree, E. Scaling up digital circuit computation with DNA strand displacement cascades. *Science 332*, 6034 (2011), 1196–1201.

[78] Qian, L., Winfree, E., and Bruck, J. Neural network computation with DNA strand displacement cascades. *Nature 475*, 7356 (2011), 368–372.

[79] Rettner, R. DNA: Definition, structure and discovery, 2013. `http://www.livescience.com/37247-dna.html`. Accessed 24 Feb 2016.

[80] Rothemund, P. W. K. Folding DNA to create nanoscale shapes and patterns. *Nature 440*, 7082 (2006), 297–302.

[81] Rothemund, P. W. K., Papadakis, N., and Winfree, E. Algorithmic self-assembly of DNA Sierpinski triangles. *PLoS Biology 2*, 12 (2004), e424.

[82] Sannohe, Y., Endo, M., Katsuda, Y., Hidaka, K., and Sugiyama, H. Visualization of dynamic conformational switching of the G-quadruplex in a DNA nanostructure. *Journal of the American Chemical Society 132*, 46 (2010), 16311–16313.

[83] Sarcar, M. M. M., Rao, K. M., and Narayan, K. L. *Computer Aided Design and Manufacturing*. PHI Learning, 2008.

[84] Seeman, N. C. DNA in a material world. *Nature 421*, 6921 (2003), 427–431.

[85] Seeman, N. C. Nanotechnology and the double helix. *Scientific American 17* (2007), 30–39.

[86] Seeman, N. C. An overview of structural DNA nanotechnology. *Molecular Biotechnology 37*, 3 (2007), 246–257.

[87] Seeman, N. C. Nanomaterials based on DNA. *Annual Review of Biochemistry 79* (2010), 65–87.

[88] Seeman, N. C. Structural DNA nanotechnology: Growing along with Nano Letters. *Nano Letters 10*, 6 (2010), 1971–1978.

[89] Selmi, D. N., Adamson, R. J., Attrill, H., Goddard, A. D., Gilbert, R. J. C., Watts, A., and Turberfield, A. J. DNA-templated protein arrays for single-molecule imaging. *Nano Letters 11*, 2 (2011), 657–660.

[90] Sequin software. Sequin – a DNA sequence submission and update tool. `http://www.ncbi.nlm.nih.gov/projects/Sequin/index.html`. Accessed 2 Sep 2016.

[91] Service, R. F. DNA nanotechnology grows up. *Science 332*, 6034 (2011), 1140–1143.

[92] Shen, B., Linko, V., Dietz, H., and Toppari, J. J. Dielectrophoretic trapping of multilayer DNA origami nanostructures and DNA origami-induced local destruction of silicon dioxide. *Electrophoresis 36*, 2 (2015), 255–262.

[93] Shen, B., Linko, V., Tapio, K., Kostiainen, M. A., and Toppari, J. J. Custom-shaped metal nanostructures based on DNA origami silhouettes. *Nanoscale 7*, 26 (2015), 11267–11272.

[94] Shen, B., Tapio, K., Linko, V., Kostiainen, M. A., and Toppari, J. J. Metallic nanostructures based on DNA nanoshapes. *Nanomaterials 6*, 8 (2016), 146.

[95] Sobczak, J.-P. J., Martin, T. G., Gerling, T., and Dietz, H. Rapid folding of DNA into nanoscale shapes at constant temperature. *Science 338*, 6113 (2012), 1458–1461.

[96] Stahl, E., Martin, T. G., Praetorius, F., and Dietz, H. Facile and scalable preparation of pure and dense DNA origami solutions. *Angewandte Chemie Internation Edition 53*, 47 (2014), 12735–12740.

[97] Tarini, M., Cignoni, P., and Montani, C. Ambient Occlusion and Edge Cueing for Enhancing Real Time Molecular Visualization. *IEEE Transactions on Visualization and Computer Graphics 12*, 5 (2006), 1237–1244.

[98] Veneziano, R., Ratanalert, S., Zhang, K., Zhang, F., Yan, H., Chiu, W., and Bathe, M. Designer nanoscale DNA assemblies programmed from the top down. *Science 352*, 6293 (2016), 1534. Software available at: `http://daedalus-dna-origami.org/`. Accessed 11 Aug 2016.

[99] Wang, P., Hyeon Ko, S., Tian, C., Hao, C., and Mao, C. RNA-DNA hybrid origami: folding of a long RNA single strand into complex nanostructures using short DNA helper strands. *Chemical Communications 49*, 48 (2013), 5462–5464.

[100] Watson, J. D., and Crick, F. H. C. Molecular structure of nucleic acids: A structure for deoxyribose nucleic acid. *Nature 171*, 4356 (1953), 737–738.

[101] Wei, B., Dai, M., and Yin, P. Complex shapes self-assembled from single-stranded DNA tiles. *Nature 485*, 7400 (2012), 623–626.

[102] Wei, R., Martin, T. G., Rant, U., and Dietz, H. DNA origami gatekeepers for solid-state nanopores. *Angewandte Chemie International Edition 51*, 20 (2012), 4864–4867.

[103] Williams, S., Lund, K., Lin, C., Wonka, P., Lindsay, S., and Yan, H. *DNA Computing: 14th International Meeting on DNA Computing, DNA 14, Prague, Czech Republic, June 2-9, 2008. Revised Selected Papers*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, ch. Tiamat: A Three-Dimensional Editing Tool for Complex DNA

Structures, pp. 90–101. Software available at: `http://yanlab.asu.edu/Resources.html`. Accessed 13 May 2016.

[104] Winfree, E., Liu, F., Wenzler, L. A., and Seeman, N. C. Design and self-assembly of two-dimensional DNA crystals. *Nature 394*, 6693 (1998), 539–544.

[105] Yin, P., Choi, H. M. T., Calvert, C. R., and Pierce, N. A. Programming biomolecular self-assembly pathways. *Nature 451*, 7176 (2008), 318–322.

[106] Zadeh, J. N., Steenberg, C. D., Bois, J. S., Wolfe, B. R., Pierce, M. B., Khan, A. R., Dirks, R. M., and Pierce, N. A. NUPACK: analysis and design of nucleic acid systems. *Journal of Computational Chemistry 32*, 1 (2011), 170–173.

[107] Zhang, D. Y., and Seelig, G. Dynamic DNA nanotechnology using strand-displacement reactions. *Nature Chemistry 3*, 2 (2011), 103–113.

[108] Zhang, F., Jiang, S., Wu, S., Li, Y., Mao, C., Liu, Y., and Yan, H. Complex wireframe DNA origami nanostructures with multi-arm junction vertices. *Nature Nanotechnology 10*, 9 (2015), 779–784.

[109] Zhang, F., Nangreave, J., Liu, Y., and Yan, H. Structural DNA nanotechnology: State of the art and future perspective. *Journal of the American Chemical Society 136*, 32 (2014), 11198–11211.

[110] Zheng, J., Birktoft, J. J., Chen, Y., Wang, T., Sha, R., Constantinou, P. E., Ginell, S. L., Mao, C., and Seeman, N. C. From molecular to macroscopic via the rational design of a self-assembled 3D DNA crystal. *Nature 461*, 7260 (2009), 74–77.

[111] Zhirnov, V., Zadegan, R. M., Sandhu, G. S., Church, G. M., and Hughes, W. L. Nucleic acid memory. *Nature Materials 15*, 4 (2016), 366–370.

[112] Zuker, M. Mfold web server for nucleic acid folding and hybridization prediction. *Nucleic Acids Research 31*, 13 (2003), 3406–3415.